# Privacy Policy Specification and Audit in a Fixed-Point Logic

- How to enforce HIPAA, GLBA and all that

Henry DeYoung, Deepak Garg, Limin Jia, Dilsun Kaynar, Anupam Datta

May 11, 2010

CMU-CyLab-10-008

CyLab
Carnegie Mellon University
Pittsburgh, PA 15213

# Privacy Policy Specification and Audit in a Fixed-Point Logic

— How to enforce HIPAA, GLBA and all that

Henry DeYoung Deepak Garg Limin Jia Dilsun Kaynar Anupam Datta

May 12, 2010

**Abstract**

Organizations such as hospitals and banks that collect and use personal information are required to comply with privacy regulations like the Health Insurance Portability and Accountability Act (HIPAA) and the Gramm-Leach-Bliley Act (GLBA). With the goal of specification and enforcement of such practical policies, we develop the logic PrivacyLFP, whose syntax is an extension of the fixed point logic LFP with operators of linear temporal logic. We model organizational processes by assigning role-based responsibilities to agents that are also expressed in the same logic. To aid in designing such processes, we develop a semantic locality criterion to characterize responsibilities that agents (or groups of agents) have a strategy to discharge, and easily checkable, sound syntactic characterizations of responsibilities that meet this criterion. Policy enforcement is achieved through a combination of techniques: (a) a design-time analysis of the organizational process to show that the privacy policy is respected if all agents act responsibly, using a sound proof system we develop for PrivacyLFP; and (b) a posthoc audit of logs of organizational activity that identifies agents who did not live up to their responsibilities, using a model checking procedure we develop for PrivacyLFP. We illustrate these enforcement techniques using a representative example of an organizational process.

## 1 Introduction

Privacy is an important concern for organizations that collect and use personal information, such as hospitals, clinics, banks, credit card clearing houses, customer support centers, and academic institutions. These organizations face the growing challenge of managing privacy risks and compliance requirements. In fact, designing organizational processes to manage personal data and ensure compliance with regulations such as the Health Insurance Portability and Accountability Act (HIPAA) and the Gramm-Leach-Bliley Act (GLBA) [32, 33] has become one of the greatest challenges facing organizations today (see, for example, a recent survey from Deloitte and the Ponemon Institute [19]). This paper develops theoretically well-founded methods to support the compliance process and presents case studies that demonstrate that the methods apply to real privacy regulations.

Our first set of contributions pertain to privacy policy specification. We present the logic PrivacyLFP (see Section 2), whose syntax is an extension of the fixed point logic LFP with operators of linear temporal logic [26]. The formulas of the logic are interpreted over traces containing agent actions, which model, for example, how agents transmit and use personal information. This logic can express common privacy policy idioms, such as conditions on retransmission of information, obligations, notifications, opt-in/opt-out options and disclosure purposes. The choice of the logic was guided by a comprehensive study of the privacy-relevant sections of the HIPAA and GLBA regulations. Specifically, in examining GLBA, we found clauses that required the use of fixed points to specify; clauses in both regulations necessitated the use of temporal operators, real-time, and disclosure purposes. This report focuses on the logic and enforcement of policies represented in it; formalization of all operational clauses of HIPAA and GLBA is contained in a separate report [20].

Our second set of contributions pertain to modeling organizational processes (see Section 4). We model organizational processes by assigning role-based responsibilities to agents. These responsibilities are also expressed in the same logic. The goal in designing processes is to ensure that if all agents act responsibly, then the policy is satisfied in every execution. However, it is important to ensure that an agent can, in fact, discharge her responsibilities. We present examples of responsibilities in PrivacyLFP that can never be discharged, and then go on to provide a semantic definition of locally feasible responsibilities, which is intended to capture "reasonable" responsibilities. To aid in designing organizational processes, we also present easily checkable, sound syntactic characterizations of responsibilities that meet this criterion, associated strategies for discharging such responsibilities, and theorems about the composition of such responsibilities (Theorem 4.2).

Our final set of contributions pertain to policy enforcement (Section 5). Policy enforcement is achieved through two logic-based methods for enforcing privacy policies. Our first method answers the question: Does a given organizational process respect a given privacy policy? This method is based on a sound *proof system* for PrivacyLFP and is described in Section 5.1. The proof system is obtained by adapting previous proof systems for an intuitionistic logic with fixed-points, $\mu$LJ [8, 17], to our classical logic PrivacyLFP; the soundness proof for the proof system with respect to the trace semantics is a new technical result. Our second enforcement method audits logs of organizational activity for violations of principals' assigned responsibilities. It is based on a novel tableau-based *model checking* procedure for PrivacyLFP that we develop and prove sound in Section 5.2. We illustrate these enforcement techniques using a representative example of an organizational process.

The approach taken in this paper builds on *contextual integrity*, a conceptual framework for understanding privacy expectations and their implications developed in the literature on law, public policy, and political philosophy [27]. The primary tenet of contextual integrity is that people interact in society not simply as individuals in an undifferentiated social world, but as individuals in certain capacities or roles, in distinctive social contexts (e.g., health care or banking). The semantic model over which the formulas of PrivacyLFP are interpreted formalizes this intuition, in a manner that is similar to prior work by Barth et al. [10, 11]. The conceptual factoring of policy enforcement into design-time analysis assuming agents are responsible and posthoc auditing for responsibility violations also originated in those papers. The results of this paper push forward the program of practical privacy policy specification and enforcement significantly by developing a first-order logic with fixed-points that has the additional expressiveness needed to specify real privacy regulations in their entirety (all privacy-relevant clauses of HIPAA and GLBA), and new enforcement techniques based on proof-theory and auditing that work for the entire logic. In contrast, the auditing procedure in Barth et al. [11] only works for a very restricted class of "graph-based workflows" and design-time analysis is achieved for a less expressive propositional fragment of a temporal logic. A more detailed comparison with prior work appears in Section 6. Concluding remarks and directions for future work appear in Section 7.

# 2 Policy Specification

We formally represent privacy laws and responsibilities as formulas of a new logic PrivacyLFP. PrivacyLFP is an extension of the logic LFP [13, 28] with temporal operators, and is interpreted against traces. LFP contains first-order quantifiers and allows definitions of predicates as greatest and least fixed-points. After motivating the need for fixed-points in formalizing privacy regulation, we briefly review LFP and its semantics (Section 2.1). Then we introduce PrivacyLFP's trace-based model (Section 2.2) and its syntax (Section 2.3). Prior work on which this paper builds [9–11] uses a different logic LPU (Logic of Privacy and Utility), which is based on alternating-time temporal logic or ATL [3]. Although LPU suffices to express representative examples of privacy regulations considered in prior work, it does not suffice to represent entire privacy laws like HIPAA and GLBA [32, 33].

Specifically, LFP and PrivacyLFP can, but LPU cannot, express predicates defined as *fixed-points* of equations, which are needed to formalize GLBA. To understand the need for fixed-points consider §6802(c) of GLBA:

> Except as otherwise provided in this subchapter, a nonaffiliated third party that receives from a financial institution nonpublic personal information under this section shall not, directly

or through an affiliate of such receiving third party, disclose such information to any other person that is a nonaffiliated third party of both the financial institution and such receiving third party, unless such disclosure would be lawful if made directly to such other person by the financial institution.

Suppose that in an attempt to formalize this clause in logic, we define the predicate $\text{maysend}(p_1, p_2, m)$ to mean that entity $p_1$ may send information $m$ to entity $p_2$. Then, roughly, the above clause would be formalized by the definition below. ($\triangleq$ denotes a definition, $\supset$ denotes implication, $\text{activerole}(p, r)$ means that principal $p$ is active in role $r$, and $\diamondminus\phi$ means that $\phi$ holds in the past.)

$$\begin{aligned}\text{maysend}(p_1, p_2, m) \triangleq \forall p'.\ & \neg\text{activerole}(p_1, \mathit{affiliate}(p')) \wedge \neg\text{activerole}(p_2, \mathit{affiliate}(p')) \wedge \\ & \neg\text{activerole}(p_2, \mathit{affiliate}(p_1)) \wedge \\ & (\diamondminus(\text{send}(p', p_1, m) \wedge \text{activerole}(p', \mathit{institution})) \supset \diamondminus\text{maysend}(p', p_2, m))\end{aligned}$$

This definition is recursive because the predicate maysend reappears in the last line on the right side of the definition. Such recursive definitions cannot be expressed easily in first-order logic or LPU. However, in LFP such definitions can be represented either using the least-fixed point operator, $\mu$, or using the greatest-fixed point operator, $\nu$, as is known from prior work [24]. In this case, the definition should correspond to the greatest fixed point since we do not want to impose any constraints on transmission beyond those stated in the body of the law. (A further explanation of this point appears with a precise formalization of this clause in Section 3.)

## 2.1 The Logic LFP

We review the syntax and semantics of the logic LFP (Least Fixed-Point Logic) limiting our discussion to the minimum necessary to explain our technical ideas; theory of the logic may be found in prior work [13, 28]. LFP is an extension of first-order logic with the least fixed-point operator $(\mu X(\vec{x}).\varphi)(\vec{t})$ and the greatest fixed-point operator $(\nu X(\vec{x}).\varphi)(\vec{t})$. The former defines an implicit predicate $X$ as the least solution of the equation $X(\vec{x}) \triangleq \varphi$ and checks that the tuple of terms $\vec{t}$ satisfies the predicate (i.e, it lies in the least solution). Both $X$ and $\vec{x}$ are in scope in $\varphi$ and can be tacitly renamed. $(\nu X(\vec{x}).\varphi)(\vec{t})$ is similar, except that it defines the predicate as the greatest solution of the same equation. The syntax of LFP formulas $\varphi, \psi$ is shown below. $t$ denotes a first-order term structure, $x, y$ are first-order variables that range over terms, $P$ denotes a predicate with a fixed interpretation, and variables $X, Y$ denote predicates defined implicitly as fixed-points.

$$\varphi, \psi \quad ::= \quad P(\vec{t}) \mid X(\vec{t}) \mid \top \mid \bot \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \neg\varphi \mid \forall x.\varphi \mid \exists x.\varphi \mid (\mu X(\vec{x}).\varphi)(\vec{t}) \mid (\nu X(\vec{x}).\varphi)(\vec{t})$$

We define implication $\varphi \supset \psi$ as $(\neg\varphi) \vee \psi$. The logic is multi-sorted, although we elide the details of sorts here. (Details of sorts relevant to formalization of HIPAA and GLBA may be found in the companion report [20].) In order to ensure that the least and greatest fixed-points always exist, any occurrences of $X$ in $\varphi$ in $(\mu X(\vec{x}).\varphi)(\vec{t})$ and $(\nu X, \vec{x}.\varphi)(\vec{t})$ must be under an even number of negations. The existence of the least and greatest solutions is then a straightforward consequence of the Knaster-Tarski theorem [31].

The semantics of LFP are based on those of first-order logic, with added provision for the fixed-point operators. Let D be an algebra matching the signature of terms and predicates of the logic, let $[\![ t ]\!]^\theta$ denote the interpretation of the term $t$ with evaluation (partial map from first-order variables to D) $\theta$ for its free first-order variables and some implicit interpretation of function symbols, and let $[\![ \vec{t} ]\!]^\theta$ be its component-wise lifting to tuples. Let $\mathcal{I}$ denote a map from predicates symbols and predicate variables free in $\varphi$ to relations of respective arities over the domain D. The semantics of a formula $\varphi$ are captured by the relation $\theta; \mathcal{I} \models \varphi$, defined by induction on $\varphi$:

$\theta; \mathcal{I} \models P(\vec{t})$ iff $[\![ \vec{t} ]\!]^\theta \in \mathcal{I}(P)$
$\theta; \mathcal{I} \models X(\vec{t})$ iff $[\![ \vec{t} ]\!]^\theta \in \mathcal{I}(X)$
$\theta; \mathcal{I} \models \top$
$\theta; \mathcal{I} \not\models \bot$

$\theta; \mathcal{I} \models \varphi \wedge \psi$ iff $\theta; \mathcal{I} \models \varphi$ and $\theta; \mathcal{I} \models \psi$
$\theta; \mathcal{I} \models \varphi \vee \psi$ iff $\theta; \mathcal{I} \models \varphi$ or $\theta; \mathcal{I} \models \psi$
$\theta; \mathcal{I} \models \neg\varphi$ iff $\theta; \mathcal{I} \not\models \varphi$.
$\theta; \mathcal{I} \models \forall x.\varphi$ iff for all $d \in \mathrm{D}$, $(\theta[x \mapsto d]; \mathcal{I} \models \varphi)$
$\theta; \mathcal{I} \models \exists x.\varphi$ iff for some $d \in \mathrm{D}$, $(\theta[x \mapsto d]; \mathcal{I} \models \varphi)$
$\theta; \mathcal{I} \models (\mu X(\vec{x}).\varphi)(\vec{t})$ iff $[\![\vec{t}]\!]^{\theta} \in \mu F^{X,\vec{x}}_{\mathcal{I},\theta}(\varphi)$
$\theta; \mathcal{I} \models (\nu X(\vec{x}).\varphi)(\vec{t})$ iff $[\![\vec{t}]\!]^{\theta} \in \nu F^{X,\vec{x}}_{\mathcal{I},\theta}(\varphi)$

In the last two clauses, $F^{X,\vec{x}}_{\mathcal{I},\theta}(\varphi) : 2^{\mathrm{D}^{|\vec{x}|}} \to 2^{\mathrm{D}^{|\vec{x}|}}$ is the function that maps a set $S$ of tuples, each with $|\vec{x}|$ components, to $\{\vec{d} \mid \theta[\vec{x} \mapsto \vec{d}]; \mathcal{I}[X \mapsto S] \models \varphi\}$. This is a monotone map because of the constraint that every occurrence of $X$ in $\varphi$ be under an even number of negations. So its greatest and least fixed points, $\nu F^{X,\vec{x}}_{\mathcal{I},\theta}(\varphi)$ and $\mu F^{X,\vec{x}}_{\mathcal{I},\theta}(\varphi)$, exist by the Knaster-Tarski theorem [31].

**Negation normal form (NNF)** For every LFP formula $\varphi$, there is a semantically equivalent formula in which negation is restricted to predicates (i.e, the form $\neg(P\ \vec{t})$). Formulas of the latter form are said to be in negation normal form or NNF. The NNF of a LFP formula $\varphi$ can be obtained by commuting negations inwards with other connectives through the DeMorgan's laws, e.g, $\neg(\psi_1 \wedge \psi_2)$ is equivalent to $(\neg\psi_1) \vee (\neg\psi_2)$. Importantly, fixed-points $\mu$ and $\nu$ are duals of each other: $\neg((\mu X(\vec{x}).\varphi)(\vec{t}))$ is equivalent to $(\nu X(\vec{x}).\neg\varphi\{\neg X/X\})(\vec{t})$. The existence of equivalent NNF formulas for all of LFP is important because one of our enforcement techniques (model-checking; Section 5.2) works only with NNF formulas. Note that the NNF formula obtained by applying DeMorgan's laws in this manner cannot have a subformula of the form $\neg(X\ \vec{t})$ because of the monotonicity requirement for predicate variables $X$ bound by $\mu$ and $\nu$ operators.

## 2.2 Traces, First-Order Structure, and Time

Next, we introduce a trace-based model for interpreting formulas of LFP. A salient feature of the model is the association of real time with states, which is necessary to express several clauses from both HIPAA and GLBA.

**Traces** Our execution model consists of several *agents or principals* $p, q$ in changing *roles* $r$, performing actions concurrently, resulting in a finite sequence of states $s_0 s_1 \ldots s_n$, also called a *trace* $\sigma$. Each state $s_{i+1}$ is derived from the previous state $s_i$ through a stipulated transition relation $s \xrightarrow{a(s)} s'$, where $a(s)$ describes the actions performed by the various agents in state $s$. A state $s$ is a tuple $(\kappa(s), \rho^A(s), \rho^B(s), a(s), \tau(s), \iota(s))$. Briefly, $\kappa(s)$ maps each agent to its knowledge of private information (a formal description of knowledge is omitted here – see the related technical report [20] for details); $\rho^A(s)$ is a function that maps each agent to the role in which it is active in state $s$; $\rho^B(s)$ is a function from agents to sets of roles that specifies the potential roles in which each agent may be active in future; $a(s)$ is the set of actions performed by agents in state $s$ that cause a transition to the next state on the trace; $\tau(s)$ is the time point associated with the state (described in detail below); and $\iota(s)$ is an interpretation of predicates in state $s$ that maps each predicate symbol $P$ in the signature of the logic to a set of tuples of terms over a domain D. D must include, at the least, principals, roles, time points, attributes and purposes (explained in Section 3), and messages that agents may send to each other.

**Interpretation on traces** To interpret formulas of LFP over traces, we restrict ourselves to a fragment of the logic in which the first argument of every atomic formula is the state in which the formula is to be interpreted, so each atomic formula has the form $P(s, \vec{t})$ or $X(s, \vec{t})$. Given a trace $\sigma$, we define the interpretation $\mathcal{I}_\sigma$ by saying that $(s, \vec{d}) \in \mathcal{I}_\sigma(P)$ if and only if $\vec{d} \in \iota(s)(P)$. Finally, we define $\theta; \sigma \models \varphi$ to mean $\theta; \mathcal{I}_\sigma \models \varphi$ (the latter relation was defined in Section 2.1). This approach to interpreting formulas against traces by making state explicit in formulas is inspired by work on hybrid modal logics [12, 14, 16]. It differs from semantic relations in temporal logic where formulas do not explicitly mention state but the semantic relation takes the state as an argument (it has the form $\theta; \sigma; s \models \varphi$). Our approach is more expressive than temporal logic because it allows us to compare states and check their properties through predicates in the logic. The predicate $s \leq_{\mathsf{st}} s'$ means that state $s$ occurs before state $s'$ in the trace of interpretation, while the function $\mathrm{next}(s)$ returns the state following $s$.

**Real Time** Privacy laws, including HIPAA and GLBA, often contain references to durations of real time. To represent wall-clock time in the logic, we follow prior work by Alur and Henzinger [2] and assume that each state $s$ is associated with a time point $\tau(s)$, which can be obtained in the logic through the function symbol $\mathrm{time}(s)$. We assume standard operators $<, +, -$, etc. on time points and require that for consecutive states $s_i$ and $s_{i+1}$ on a trace, $\mathrm{time}(s_i) < \mathrm{time}(s_{i+1})$. As a result, $s <_{\mathsf{st}} s'$ in the logic if and only if $\mathrm{time}(s) < \mathrm{time}(s')$. To make it easier to access the wall-clock time, we include the so-called freeze quantifier $\downarrow x.\phi$ of TPTL in PrivacyLFP (Section 2.3). $\downarrow x.\phi$ binds the time of interpretation to $x$ in $\phi$. Several examples of the use of real time in privacy laws are presented in Section 3.

### 2.3 PrivacyLFP: LFP + Temporal Operators

The logic PrivacyLFP consists of an expanded syntax for LFP and is interpreted over the model defined in Section 2.2 through a translation to LFP, which we present in this section. The need for an expanded syntax is motivated by two reasons. First, the expanded syntax includes several operators of linear time temporal logic (LTL) [26] as well as the freeze quantifier $\downarrow x.\varphi$ of Alur and Henzinger [2], all of which make it easier to represent time and relative order of events in privacy policies. Second, the expanded syntax elides the need to list the state of interpretation explicitly in each predicate (which we introduced in Section 2.2 to allow interpretation of formulas on traces), because its translation to LFP is parametrized by the state of interpretation and embeds that state as the first argument of each atomic formula automatically.

Formulas in PrivacyLFP are denoted $\phi, \psi$. They include all connectives of LFP, standard linear temporal logic operators: $\Diamond\phi$ ($\phi$ holds in some future state), $\diamondminus\phi$ ($\phi$ holds in some past state), $\Box\phi$ ($\phi$ holds in every future state), $\boxminus\phi$ ($\phi$ holds in every past state), $\mathbf{G}\ \phi$ ($\phi$ holds in every state), $\phi\,\mathcal{U}\,\psi$ ($\psi$ holds eventually and $\phi$ holds until then), $\phi\,\mathcal{S}\,\psi$ ($\psi$ held in the past and $\phi$ holds since then), and $\phi\,\mathcal{W}\,\psi$ ($\phi$ holds forever or until $\psi$ holds) as well the "freeze" quantifier $\downarrow x.\phi$ which binds to $x$ in $\phi$ the *time* of interpretation. The meaning of PrivacyLFP formulas is defined by the function $(\phi)^{@s}$ which translates, at state $s$, a formula $\phi$ in PrivacyLFP to a formula in LFP.

$$
\begin{array}{rcl}
(P(\vec{t}))^{@s} & \triangleq & P(s, \vec{t}) \\
((\nu X(\vec{x}).\phi)(\vec{t}))^{@s} & \triangleq & (\nu X(y, \vec{x}).\phi^{@y})(s, \vec{t}) \\
(\Diamond\phi)^{@s} & \triangleq & \exists s'.\,(s \leq_{\mathsf{st}} s') \wedge \phi^{@s'} \\
(\diamondminus\phi)^{@s} & \triangleq & \exists s'.\,(s' \leq_{\mathsf{st}} s) \wedge \phi^{@s'} \\
(\Box\phi)^{@s} & \triangleq & \forall s'.\,(s \leq_{\mathsf{st}} s') \supset \phi^{@s'} \\
(\boxminus\phi)^{@s} & \triangleq & \forall s'.\,(s' \leq_{\mathsf{st}} s) \supset \phi^{@s'} \\
(\mathbf{G}\ \phi)^{@s} & \triangleq & \forall s'.\,\phi^{@s'} \\
(\phi\,\mathcal{U}\,\psi)^{@s} & \triangleq & \exists s'.\,(s \leq_{\mathsf{st}} s') \wedge \psi^{@s'} \wedge (\forall s''.\,(s \leq_{\mathsf{st}} s'') \wedge (s'' <_{\mathsf{st}} s') \supset \phi^{@s''}) \\
(\phi\,\mathcal{S}\,\psi)^{@s} & \triangleq & \exists s'.\,(s' \leq_{\mathsf{st}} s) \wedge \psi^{@s'} \wedge (\forall s''.\,(s' <_{\mathsf{st}} s'') \wedge (s'' \leq_{\mathsf{st}} s) \supset \phi^{@s''}) \\
(\phi\,\mathcal{W}\,\psi)^{@s} & \triangleq & (\Box\phi)^{@s} \vee (\phi\,\mathcal{U}\,\psi)^{@s} \\
(\bigcirc\phi)^{@s} & \triangleq & \phi^{@\mathrm{next}(s)} \\
(\downarrow x.\,\phi)^{@s} & \triangleq & ([\mathrm{time}(s)/x]\phi)^{@s}
\end{array}
$$

In the sequel, we represent policies and responsibilities in PrivacyLFP but owing to its definability in LFP, develop analysis methods (proof theory and model checking in Section 5) for LFP only.

## 3 Case Studies: GLBA and HIPAA

Our choice of the logic PrivacyLFP for analysis of privacy laws and policies is based on two real-life case studies wherein we represent (in PrivacyLFP) all the privacy-relevant sections of the Gramm-Leach-Bliley

Act (GLBA) [32] that regulates disclosures of private information in financial institutions like banks and the Health Insurance Portability and Accountability Act (HIPAA) [33] that regulates protected health information. To the best of our knowledge, these are the most complete formalizations of GLBA or HIPAA in a formal logic or language to date. Both our case studies are substantial: the encoding of GLBA spans 13 pages, while that of HIPAA requires over 100 pages (both page counts include explanations of logical formulas). Although the details of these formalizations are the subject of a separate technical report [20], we briefly discuss salient points of the case studies and use examples from them to illustrate the use of PrivacyLFP in formalizing privacy regulations.

**The Gramm-Leach-Bliley Act (GLBA)** Our formalization of GLBA covers §6802 and §6803 of the law and relies on §6809 for definitions of key concepts. §6802 describes several conditions, *all of which* must hold in order for a disclosure of a client's private information by a financial institution to be considered legal. Borrowing terms from prior work on LPU, we call such conditions *negative norms*, symbolically denoted $\varphi^-$. (In contrast, positive norms $\varphi^+$ are conditions of which *any one* must hold in order for a transaction to be considered legal. GLBA does not have any positive norms but HIPAA does as we explain later.) §6803 pertains to disclosures that a financial institution must make to its clients, e.g, every financial institution must remind all customers of its privacy policies annually (§6803(a)). Finally, §6809 defines transmissions that are covered under this law. Roughly, it states that even transmissions made by principals acting on behalf of a financial institution (e.g, disclosures by a financial institution's attorneys) are covered under the law. To account for this, we define a macro $\mathrm{hlsend}(p_1, p_2, m)$ which intuitively means that someone acting on behalf of $p_1$ sends message $m$ to someone acting on behalf of $p_2$ and write our formalization using this predicate instead of the expected predicate $\mathrm{send}(p_1, p_2, m)$, which means that $p_1$ sends message $m$ to $p_2$. The overall formalization of GLBA takes the form shown below. The formalization retains the structure of the law; subscripts on various $\varphi$'s are corresponding clause numbers from the text of the law. Formula $\mathrm{contains}(m, q, t)$ means that message $m$ contains information about attribute $t$ of subject $q$, e.g, $\mathrm{contains}(m, \mathit{address}, \mathrm{Alice})$ means that message $m$ contains Alice's address; $\mathrm{info}(d, u)$ is the message obtained by tagging the raw data $d$ with purpose $u$ (e.g, billing); $\mathrm{beginrole}(q, r)$ means that principal $q$ begins to belong to role $r$.

$$
\begin{array}{l}
\mathbf{G}\,((\forall p_1', p_2', m'.\, \mathrm{hlsend}(p_1', p_2', m') \supset \\
\quad (\nu \mathrm{maysend}(p_1, p_2, m). \\
\quad\quad \forall d, u, q, t.\, (m = \mathrm{info}(d, u)) \wedge \mathrm{contains}(m, q, t) \supset \varphi^-_{\mathtt{6802ae}} \wedge \varphi^-_{\mathtt{6802be}} \wedge \varphi^-_{\mathtt{6802c}} \wedge \varphi^-_{\mathtt{6802d}})(p_1', p_2', m')) \wedge \\
\hline
\quad (\forall q, p, r.\, \mathrm{beginrole}(q, r) \wedge (r = \mathit{customer}(p)) \supset \varphi^-_{\mathtt{6803a}} \vee \varphi^+_{\mathtt{6803d1}}))
\end{array}
$$

Parts of the formalization corresponding to §6802 and §6803 are separated by a horizontal line for readability. The part above the line states that $p_1'$ may send message $m'$ to $p_2'$ only if $\mathrm{maysend}(p_1', p_2', m')$ holds, where the predicate $\mathrm{maysend}(p_1, p_2, m)$ (or the permission to send) is defined recursively as a greatest fixed point over the negative norms $\varphi^-_{\mathtt{6802ae}}$ – $\varphi^-_{\mathtt{6802d}}$ of the law. The fixed-point is needed because, as discussed in Section 2, $\varphi^-_{\mathtt{6802c}}$ mentions maysend again. The part below the line formalizes §6803; it states that if principal $q$ enters into a customer relationship with financial institution $p$, then $p$ must make certain privacy-related disclosures to $q$, as codified in the norm $\varphi^-_{\mathtt{6803a}}$. The norm $\varphi^+_{\mathtt{6803d1}}$ is an *exception* to these required disclosures and is therefore marked with the opposite polarity $+$. As illustrations, we show the formulas $\varphi^-_{\mathtt{6802c}}$ and $\varphi^-_{\mathtt{6803a}}$. The former, §6802, was mentioned as the motivating example for fixed-points in Section 2. Readers may wish to revisit Section 2 for the legal text of the clause. Formula $\mathrm{activerole}(p, r)$ means that $p$ is active in the role $r$; $\mathrm{belongstorole}(p, r)$ means that $p$ is affiliated with role $r$ but may not be acting in it in the current state; and $t \in_{\mathcal{T}} \mathit{npi}$ means that attribute $t$ would generally not be public information, e.g, social-security number.

$$
\begin{array}{ll}
\varphi^-_{\mathtt{6802c}} \triangleq \forall p', m''. & \neg\mathrm{activerole}(p_1, \mathit{affiliate}(p')) \wedge \\
& (\neg\mathrm{activerole}(p_2, \mathit{affiliate}(p')) \wedge \\
& \ \neg\mathrm{activerole}(p_2, \mathit{affiliate}(p_1))) \wedge \\
& (t \in_{\mathcal{T}} \mathit{npi}) \wedge \\
& \diamondminus(\mathrm{hlsend}(p', p_1, m'') \wedge
\end{array}
$$

$$
\begin{aligned}
&\text{contains}(m'', q, t) \land \\
&\text{activerole}(p', \textit{institution}) \land \\
&\neg\text{activerole}(p_1, \textit{affiliate}(p')) \land \\
&\text{belongstorole}(q, \textit{consumer}(p'))) \supset \\
&\diamondsuit \text{maysend}(p', p_2, m'')
\end{aligned}
$$

In conjunction with the overall formula for GLBA, this norm means that principal $p_1$ has permission to send message $m$ containing attribute $t$ about $q$ (from which some *npi* or nonpublic protected information about $q$ can be inferred) to $p_2$ only if for every message $m''$ containing $(q, t)$ that $p_2$ received from some non-affiliate $p'$ in the past, it is also the case that $p'$ had permission to send $m''$ to $p_2$ directly. The fact that $p_1$'s permission to send is dependent on $p'$s permission to send is represented through the greatest-fixed point which defines the predicate maysend in the top-level formula.

The second norm we illustrate, §6803a, highlights the use of clock time. Its legal description and formalization are:

> At the time of establishing a customer relationship with a consumer and not less than annually during the continuation of such relationship, a financial institution shall provide a clear and conspicuous disclosure to such consumer [...], of such financial institution's policies and practices with respect to [disclosing nonpublic personal information].

$$
\begin{aligned}
\varphi^-_{\texttt{6803a}} \triangleq\ & (\exists m''.\ \text{hlsend}(p_1, q, m'') \land \\
&\qquad \text{is-annual-notice}(m'', p_1, q)) \land \\
&((\downarrow x.\ \diamondsuit(\downarrow y.\ (y \le x + 365) \land \\
&\qquad ((\exists m''.\ \text{hlsend}(p_1, q, m'') \land \\
&\qquad\qquad \text{is-annual-notice}(m'', p_1, q)) \lor \\
&\qquad \text{endrole}(q, \textit{customer}(p_1))))) \ \mathcal{W} \\
&\ \text{endrole}(q, \textit{customer}(p_1)))
\end{aligned}
$$

Together with the overall specification of GLBA, this norm requires that a financial firm $p$ send an annual notice of its privacy policies (represented by $m''$) to a customer when the customer establishes a relationship with $p$ and subsequently every year unless the relationship ends. Real time, expressed through the sequence of operators $\downarrow x.\ \diamondsuit(\downarrow y.\ (y \le x + 365) \land \ldots)$ ensures that, for every state $x$, there exists a state $y$ occurring no more than 365 days later in which the annual notice is sent.

**The Health Insurance Portability and Accountability Act (HIPAA)** For HIPAA, we formalize §164.502, §164.506, §164.508, §164.510, §164.512, §164.514, and §164.524 of the CFR. §164.502–§164.514 define conditions when a covered entity, which is the HIPAA abstraction for an organization or individual that handles private health information, may disclose such information to other principals. In general, disclosures are allowed for purposes of treatment, for adherence to law, and when prior consent has been obtained from the subject of the information being disclosed. §164.524 specifies rules for responding to requests for health information by patients. Overall, the top-level formula for HIPAA has the following form. (Formula req_for_access$(p_1, t)$ is a request by principal $p_1$ that attribute $t$, e.g, *lab_results*, be retrieved from $p_1$'s medical record and given to it.)

$$
\begin{aligned}
&\mathbf{G}\ (\forall p_1, p_2, m.\ \text{send}(p_1, p_2, m) \supset \\
&\qquad (\forall d, u, q.\ (m = \text{info}(d, u)) \land \text{contains}(m, q, t) \supset \textstyle\bigvee_i \varphi_i^+ \land \bigwedge_i \varphi_i^-) \land \\
&\overline{\phantom{xx}} \\
&\qquad (\forall t.\ (m = \text{req\_for\_access}(p_1, t)) \supset \varphi^-_{\texttt{164.524b2i'}} \lor \varphi^-_{\texttt{164.524b2ii'}}))
\end{aligned}
$$

$\varphi_i^+$ and $\varphi_i^-$ represent various positive and negative norms to permit disclosures defined in §164.502–§164.514. Some of these norms are listed in Figure 1. We mention three salient, high-level differences from the formalization of GLBA. First, the formalization of HIPAA does not require fixed-point operators, although it does require real time, e.g, in formulas $\varphi^-_{\texttt{164.512c2}}$ and $\varphi^-_{\texttt{164.524b2i}}$ in Figure 1. Second, as opposed

$$\varphi^{+}_{\texttt{164.502b2i}} \triangleq \mathrm{activerole}(p_2, \mathit{provider}) \wedge (u \in_{\mathcal{U}} \mathit{treatment})$$

$$\begin{aligned}\varphi^{+}_{\texttt{164.506c1}} \triangleq\ & \mathrm{activerole}(p_1, \mathit{covered\text{-}entity}) \wedge (t \in_{\mathcal{T}} \mathit{phi}) \wedge \\ & ((u \in_{\mathcal{U}} \mathit{treatment}(p_1)) \vee (u \in_{\mathcal{U}} \mathit{payment}(p_1)) \vee (u \in_{\mathcal{U}} \mathit{healthcare\text{-}operations}(p_1)))\end{aligned}$$

$$\begin{aligned}\varphi^{-}_{\texttt{164.512c2}} \triangleq\ & {\downarrow} x.\exists m'.\ (\diamondminus \mathrm{send}(p_1, q, m') \vee \Diamond({\downarrow} y.\ (y \leq x + c_{prompt}) \wedge \mathrm{send}(p_1, q, m'))) \wedge \\ & \mathrm{is\text{-}notice\text{-}of\text{-}report}(m', p_1, p_2, (q, t), u)\end{aligned}$$

($c_{prompt}$ is a time constant that captures the term "promptly" in the law's text)

$$\begin{aligned}\varphi^{-}_{\texttt{164.524b2i}} \triangleq\ & {\downarrow} x.\ \mathrm{accessible\text{-}on\text{-}site}(p_2, (p_1, t)) \supset \Diamond({\downarrow} y.\ (y \leq x + 30) \wedge (\mathrm{respond\text{-}164.524b2iA}(p_2, (p_1, t)) \vee \\ & \mathrm{respond\text{-}164.524b2iB}(p_2, (p_1, t))))\end{aligned}$$

Figure 1: Representative norms from HIPAA formalized in PrivacyLFP

to GLBA, HIPAA includes positive norms as well (e.g, $\varphi^{+}_{\texttt{164.502b2i}}$ and $\varphi^{+}_{\texttt{164.506c1}}$). These are combined disjunctively in the formalization because only one of these must be satisfied to permit a transmission. Finally, permission to disclose protected health information under HIPAA is often contingent upon the purpose (treatment, payment, health care, etc.) of the disclosure, but HIPAA does not regulate that the recipient of the information use it for exactly the intended purpose. To model this, we assume that the sender of each message lists its purpose in the message – messages with protected health information have the form $\mathrm{info}(d, u)$ in the top-level formula above where $d$ is the data content and $u$ its purpose – and allow the norms $\varphi_i^+$ and $\varphi_i^-$ to check the purpose $u$ against those mentioned in HIPAA. Examples of formulas with such checks are $\varphi^{+}_{\texttt{164.502b2i}}$ and $\varphi^{+}_{\texttt{164.506c1}}$ in Figure 1. The predicate $u \in_{\mathcal{U}} u'$ means that purpose $u$ is a specific form of purpose $u'$, e.g, $\mathit{blood\_test} \in_{\mathcal{U}} \mathit{treatment}$.

# 4 Organizational Process Model

In Section 4.1, we model organizational processes by assigning role-based responsibilities (expressed in PrivacyLFP) to agents. Specifically, we show through an example how such a model could mirror an organization's natural hierarchy. It is important to ensure that an agent can, in fact, discharge her assigned responsibilities. In Section 4.2, we present examples of responsibilities in PrivacyLFP that can never be discharged, and then go on to provide a semantic definition of locally feasible responsibilities, which can be discharged. To aid in designing organizational processes, we also present easily checkable, sound syntactic characterizations of responsibilities that meet this criterion, associated strategies, and results about composition of such responsibilities (Theorems 4.2).

## 4.1 Role-based Responsibility

We model organizational processes by assigning role-based responsibilities (expressed in PrivacyLFP) to agents. Agents can either be individuals, organizational units, or software systems (reference monitors) that aid in policy enforcement. The responsibilities of human agents can be arbitrary formulas in PrivacyLFP while responsibilities for software systems should not contain any predicates whose truth value cannot be automatically determined by looking at a trace (e.g, the predicate $\mathrm{contains}(m, q, t)$ predicate from Section 3).

We show through an example how such a model could mirror an organization's natural hierarchy. Figure 2 contains an illustration of the processes and a summary of the policies and responsibilities involved in this example. The high-level policy $\varphi_{pol}$ resembles the first half of the top-level formula of GLBA from Section 3, and contains simplified policies from GLBA. The body of the greatest fixed-point contains the conjunction of two negative norms. The first norm states that $p_1$ may send to $p_2$ message $m$, which contains information $t$ about principal $q$, only if in the past, $p_1$ has sent $q$ a notice of disclosure. The second negative norm is a simplified version of $\varphi^{-}_{\mathsf{6802c}}$, which states that if $p_1$ ever received from another principal $p'$, a message containing the same information $t$ about principal $q$, then $p_1$ may disclose this information to $p_2$ only if $p'$ is allowed to send this information directly to $p_2$.

Enforcing $\varphi_{pol}$ requires an institution to determine whether another institution may disclose information. This is not always possible since an institution may not have the ability to observe all actions performed by other principals. An alternative is to allow an institution $p_1$ to directly send a message and ask the other institution $p_2$ whether $p_2$ may disclose a certain piece of information. In turn $p_1$ will be responsible for answering similar queries about itself. Such a process can be modeled by two responsibilities, $\varphi_{r1}$ and $\varphi_{r2}$. $\varphi_{r1}$ states that $p_1$ can send a message to $p_2$ only if there was a notice of disclosure, and if $p_1$ received the information from $p'$, then $p'$ must have replied to $p_1$'s query and confirmed that $p'$ can send the message directly. $\varphi_{r2}$ requires that whenever $p_1$ replies to $p_2$, $p_1$ indeed is allowed to send the information. Notice that the conditions under which $p_1$ is allowed to reply are the same as those under which $p_1$ may disclose the information.

The picture at the top of Figure 2 illustrates the internal processes of an institution $P$. There are three departments: a disclosure department ($D$), which is in charge of sending disclosures; a main send and receive department ($SR$), which is in charge of sending and receiving messages outside $P$; and a query and reply department ($QR$), which is in charge of querying another institution whether certain information can be sent, and answering similar queries from other institutions. $SR$ decides whether a send is allowed or not by asking the disclosure department if it has sent a disclosure, and if $SR$ wants to forward a message it received from another institution, it asks $QR$ whether that institution could send that information directly.

Each of these three department is modeled by its responsibilities, which are represented using logical formulas. We selectively list some of the formulas in the figure. In the next section, we present policy enforcement techniques using which we can show that $D$, $SR$, and $QR$ departments together fulfill $P$'s responsibilities $\varphi_{r1}$ and $\varphi_{r2}$ and that if all institutions fulfill their responsibilities, they collectively comply with the high-level policy $\varphi_{pol}$.

## 4.2 Locally Feasible Responsibilities

An agent should be assigned responsibilities that can be discharged using her capabilities. Typically, an agent may not be able to observe all actions of other agents, or cause another agent to perform an action. Consider the following responsibilities assigned to agent $p$:

$\varphi_1 = \forall c, m. \diamondminus \mathrm{send}(c, p, m) \supset \exists m'.\mathrm{send}(b, c, m')$.

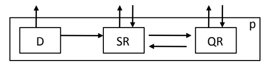
$\varphi_2 = \forall m. \Diamond \mathrm{send}(b, p, m) \supset \exists m'.\mathrm{send}(p, c, m')$

$\varphi_1$ requires an agent $b$ to send a message to $c$ if $c$ has sent a message to $p$. $p$ cannot fulfill this responsibility because she does not have the power to cause $b$ to send a message. $\varphi_2$ requires $p$ to send a message $m'$ to $c$ if in the future, $b$ sends a message to $p$. $p$ cannot fulfill this responsibility because she cannot predict the future.

Intuitively, a reasonable responsibility for an agent $p$ has to be *local* in the sense that it only depends on histories of the system execution observable by $p$, and it has to be *feasible* in the sense that $p$ has a strategy to fulfill it using only her own actions. We make this intuition precise in the following definitions.

**Definitions** To talk about $p$'s plans to fulfill her responsibilities, we define *planned traces* $\widehat{\sigma}$, which contain planned inactions. We write $\neg a$ to denote a function that map states to a set of inactions in that state. A state in a planned trace is a tuple $(\kappa(s), \rho(s), a(s), \neg a(s), \tau(s), \iota(s))$. If $\mathrm{send}(p, q, m) \in \neg a(s)$, then in state $s$, $p$ does not send $q$ message $m$. Other elements in $\widehat{\sigma}$ have the same meaning as those in ordinary traces (Section 2.2).

A planned trace $\widehat{\sigma}$ is well formed if $a(s) \cap \neg a(s) = \emptyset$. It is important to include these inactions in $\widehat{\sigma}$ because they help us detect inconsistencies between an agent's plans in different states. We define a function $Tr(\widehat{\sigma})$ to convert $\widehat{\sigma}$ to a normal trace. This function simply erases all parts of the trace associated with inactions.




Overall privacy policy

$$\begin{aligned}\varphi_{pol} = \ & \mathbf{G}\forall p_1', p_2', m'.\, \text{hlsend}(p_1', p_2', m') \supset \\ & (\nu \text{maysend}(p_1, p_2, m). \\ & \forall q, t.\, \text{contains}(m, q, t) \supset \\ & \quad (\diamondminus \text{hlsend}(p_1, q, f\_dis(p_1, p_2, q, t) \wedge \\ & \quad \forall p', m''.\text{hlsend}(p', p_1, m'') \wedge \text{contains}(m'', q, t) \supset \\ & \quad \diamondminus \text{maysend}(p', p_2, m''))(p_1', p_2', m'))\end{aligned}$$

Responsibilities for an institution (note: there is no use of fixed-points)

$$\begin{aligned}\varphi_{r1} = \ & \mathbf{G}\ \forall p_1, p_2, m.\, \text{hlsend}(p_1, p_2, m) \supset \\ & \forall q, t.\, \text{contains}(m, q, t) \supset \\ & (\diamondminus \text{hlsend}(p_1, q, f\_dis(p_1, p_2, q, t)) \wedge \\ & \forall p', m'.\text{hlsend}(p', p_1, m') \wedge \text{contains}(m', q, t) \supset \\ & \quad \diamondminus \text{send}(p', p_1, f\_reply\_maysend(p', p_2, m')))\end{aligned}$$

$$\begin{aligned}\varphi_{r2} = \ & \mathbf{G}\ \forall p_1, p_0, p_2, m.\, \text{send}(p_1, p_0, f\_reply\_maysend(p_1, p_2, m)) \supset \\ & \forall q, t. \text{contains}(m, q, t) \supset \\ & (\diamondminus \text{hlsend}(p_1, q, f\_dis(p_1, p_2, q, t)) \wedge \\ & \forall p', m'.\text{hlsend}(p', p_1, m') \wedge \text{contains}(m', q, t) \supset \\ & \quad \diamondminus \text{send}(p', p_1, f\_reply\_maysend(p', p_2, m')))\end{aligned}$$

Responsibilities for $P$'s internal departments

$$\begin{aligned}\varphi_D = \ & \mathbf{G}\forall p', m, q, t.\, \text{send}(D, SR, f\_sent\_dis(p', q, t) \supset \\ & \diamondminus \text{hlsend}(D, q, f\_dis(p, p', q, t))\end{aligned}$$

$$\begin{aligned}\varphi_{SR1} = \ & \mathbf{G}\forall p_2, m, q, t.\, \text{hlsend}(SR, p_2, m) \wedge \text{contains}(m, q, t) \supset \\ & (\diamondminus \text{send}(D, SR, f\_sent\_dis(p_2, q, t)) \wedge \\ & \forall p', m'.\text{hlsend}(p', SR, m') \wedge \text{contains}(m', q, t) \supset \\ & \diamondminus \text{send}(QR, SR, f\_reply\_maysendI(p', p_2, m')))\end{aligned}$$

$$\begin{aligned}\varphi_{SR2} = \ & \forall p_1, m.\, \text{send}(SR, QR, f\_maysend(p_1, m) \supset \\ & (\diamondminus \text{send}(D, SR, f\_sent\_dis(p_1, q, t)) \wedge \\ & \forall p', m'.\text{hlsend}(p', SR, m') \wedge \text{contains}(m', q, t) \supset \\ & \diamondminus \text{send}(QR, SR, f\_reply\_maysendI(p', p_1, m')))\end{aligned}$$

$$\begin{aligned}\varphi_{QR1} = \ & \mathbf{G}\forall p_1, p_2, m.\, \text{send}(QR, SR, f\_reply\_maysendI(p_1, p_2, m)) \supset \\ & \diamondminus \text{send}(p_1, QR, f\_reply\_maysend(p_1, p_2, m))\end{aligned}$$

$$\begin{aligned}\varphi_{QR2} = \ & \forall p_1, p_2, m.\, \text{send}(QR, p_1, f\_reply\_maysend(p, p_2, m) \supset \\ & (\diamondminus \text{send}(SR, QR, f\_maysend(p_2, m))\end{aligned}$$

Functions such as $f\_dis(p_1, p_2, q, t)$ generate a particular kind of message. For instance, $f\_dis(p_1, p_2, q, t)$ is a notice of disclosure to $q$ stating that $p_1$ will disclose to $p_2$ information $t$.

Figure 2: Example Process

We say $q$ is the *performer* of $P\ \vec{t}$ if $q$ can perform the action represented by predicate $P\ \vec{t}$. For example, $p$ is the performer of $\text{send}(p, q, m)$.

We write $\widehat{\sigma}\,|_i$ to denote the projection of $\widehat{\sigma}$ up to the $i^{th}$ state. We write $\widehat{\sigma}\,|_{i-1}= \emptyset$ to mean that the domain of all functions defining $\widehat{\sigma}$ does not contain any state that is earlier than state $i$.

We write $\widehat{\sigma}\,|_S^{\mathcal{A}}$ to denote the trace that is the same as $\widehat{\sigma}$ except that it only contains actions and inactions of which an agent in $S$ is a performer. We write $\widehat{\sigma}\,|_S^{\mathcal{V}}$ to denote the trace containing only the parts of $\widehat{\sigma}$ that are observable by some agent in $S$.

We say $i$ is in the domain of $\widehat{\sigma}$ if $i$ is in the domain of all the functions that define $\widehat{\sigma}$.

To ensure composition, we assume that all agents agree on the smallest increment of time interval $\imath$, and we only consider traces where for each state $i$ and $i+1$, $\tau(i+1) = \tau(i) + \imath$. A starting time $t_0$ for state 0 uniquely determines the time points for the rest of the states. Given $\tau$, we write $\mathit{start}(\tau)$ to denote the time point at state 0. We say $\tau$ is compatible with $\tau'$ if $\mathit{start}(\tau) = \mathit{start}(\tau')$.

We write $\widehat{\sigma}_1 \uplus \widehat{\sigma}_2$ to denote the merge of $\widehat{\sigma}_1$ and $\widehat{\sigma}_2$. Intuitively, it is obtained by taking the union of knowledge map, role sets, action sets and predicates of states that have the same timestamp. Let us use subscripts 1 to index functions defining $\widehat{\sigma}_1$ and 2 to index those defining $\widehat{\sigma}_2$. The merge operation is well defined only if for all states $i$ that are in the domain of both $\widehat{\sigma}_1$ and $\widehat{\sigma}_2$, i.e, for $i$ that satisfy:

- $\tau_1(i) = \tau_2(i)$
- $\rho_1^A(i) = \rho_2^A(i)$
- $a_1(i) \cap \neg a_2(i) = \emptyset$, $a_2(i) \cap \neg a_1(i) = \emptyset$

**Def.** $\varphi_r$ is local to a set of agents $Sa$ if for all traces $\sigma_1$ and $\sigma_2$, $\sigma_1\,|_{Sa}^{\mathcal{V}}= \sigma_2\,|_{Sa}^{\mathcal{V}}$ implies $\sigma_1 \vDash \varphi_r$ iff $\sigma_2 \vDash \varphi_r$.

$\varphi_r$ is local to a set of agents $Sa$ if it does not depend on states that are not observable by any agent in $Sa$.

The responsibilities we focus on have the form $\mathbf{G}\ \varphi_r$. A set of responsibilities $\Phi$ is feasible for a group of agents $Sa$ if agents in $Sa$ collectively have a strategy to cause each $\varphi_r$ in $\Phi$ to be true at every state $i$, and future actions will not affect the validity of $\varphi_r$ at $i$.

**Def.** A set of responsibilities $\Phi$ is feasible for a set of agents $Sa$ if for all $j$ the following holds

$$
\begin{array}{l}
\forall \widehat{\sigma}_o, H(\widehat{\sigma}_0, 0, Sa, t_0) \supset \\
\quad \exists \widehat{\sigma}_0^{Sa}, F(\widehat{\sigma}_0^{Sa}, 0, Sa, t_0) \wedge \widehat{\sigma}_0 \uplus \widehat{\sigma}_0^{Sa} \text{ is well-defined} \\
\qquad \forall \widehat{\sigma}' \text{ such that } \widehat{\sigma}'\,|_0 = \emptyset \wedge \widehat{\sigma}' \uplus \widehat{\sigma}_0 \uplus \widehat{\sigma}_0^{Sa} \text{ is well-defined} \supset \\
\qquad\quad \forall\ \mathbf{G}\ \varphi_i \in \Phi,\ \mathit{Tr}(\widehat{\sigma}' \uplus \widehat{\sigma}_0 \uplus \widehat{\sigma}_0^{Sa}), 0 \vDash \varphi_i \wedge \\
\qquad\quad \forall \widehat{\sigma}_1, H(\widehat{\sigma}_1, 1, Sa, t_0) \supset \\
\qquad\qquad \exists \widehat{\sigma}_1^{Sa}, F(\widehat{\sigma}_1^{Sa}, 1, Sa, t_0) \wedge \widehat{\sigma}_0 \uplus \widehat{\sigma}_0^{Sa} \uplus \widehat{\sigma}_1 \uplus \widehat{\sigma}_1^{Sa} \text{ is well-defined} \\
\qquad\qquad\quad \forall \widehat{\sigma}' \text{ such that } \widehat{\sigma}'\,|_1 = \emptyset \wedge \widehat{\sigma}' \uplus \widehat{\sigma}_0 \uplus \widehat{\sigma}_0^{Sa} \uplus \widehat{\sigma}_1 \uplus \widehat{\sigma}_1^{Sa} \text{ is well-defined} \supset \\
\qquad\qquad\qquad \forall\ \mathbf{G}\ \varphi_i \in \Phi,\ \mathit{Tr}(\widehat{\sigma}' \uplus \widehat{\sigma}_0 \uplus \widehat{\sigma}_0^{Sa} \uplus \widehat{\sigma}_1 \uplus \widehat{\sigma}_1^{Sa}), 1 \vDash \varphi_i \wedge \\
\qquad\qquad \dots \\
\qquad\qquad\quad \forall \widehat{\sigma}_j, H(\widehat{\sigma}_j, j, Sa, t_0) \supset \\
\qquad\qquad\qquad \exists \widehat{\sigma}_j^{Sa}, F(\widehat{\sigma}_j^{Sa}, j, Sa, t_0) \wedge \\
\qquad\qquad\qquad\quad \biguplus_{k=0}^{j} \widehat{\sigma}_k \uplus \widehat{\sigma}_k^{Sa} \text{ is well-defined} \\
\qquad\qquad\qquad\quad \forall \widehat{\sigma}' \text{ such that } \widehat{\sigma}'\,|_j = \emptyset \wedge \widehat{\sigma}' \uplus \biguplus_{k=0}^{j} \widehat{\sigma}_k \uplus \widehat{\sigma}_k^{Sa} \text{ is well-defined} \supset \\
\qquad\qquad\qquad\qquad \forall\ \mathbf{G}\ \varphi_i \in \Phi,\ \mathit{Tr}(\widehat{\sigma}' \uplus \biguplus_{k=0}^{j} \widehat{\sigma}_k \uplus \widehat{\sigma}_k^{Sa}), j \vDash \varphi_i
\end{array}
$$

where $H(\widehat{\sigma}, k, Sa, t_0)$ is true when the domain of $\widehat{\sigma}$ contains only state $k$, $\widehat{\sigma}$ does not contain any actions from any agent in $Sa$ and $t_0$ is the time point for state 0 according to $\tau$. $F(\widehat{\sigma}, k, Sa, t_0)$ is true when $\widehat{\sigma}$ contains only actions from agents in $Sa$, and all the planned actions are for states no earlier than $k$, and $t_0$ is the time point for state 0 according to $\tau$. More formal definitions can be found in Figure 6.

In the above definition, $\widehat{\sigma}_i$ are actions by agents other than those in $Sa$ at state $i$, and $\widehat{\sigma}_i^{Sa}$ are actions by agents in $Sa$. The alternating $\forall$ and $\exists$ quantification ensures that given any actions by agents not in $Sa$, agents in $Sa$ have a way to cause $\varphi_i$ to be true in that state, and any future extension of the trace will not affect the validity of $\varphi_i$ at the current state. The condition that requires $\biguplus_{k=0}^{j} \widehat{\sigma}_k \uplus \widehat{\sigma}_k^{Sa}$ to be well-defined ensures that an agent's current decisions should not conflict with any of her past decisions. Finally, given any $\widehat{\sigma}'$ that only concerns states later than $j$, the merge of $\widehat{\sigma}'$ and all the planned traces by agents in $Sa$ ($\widehat{\sigma}_k^{Sa}$) and all the planned traces by agents not in $Sa$ ($\widehat{\sigma}_k$) satisfies all the responsibilities in $\Phi$ at state $j$. The use of $\widehat{\sigma}'$ is to ensure that once agents in $Sa$ have decided on a strategy at state $j$, all the responsibilities in $\Phi$ should still hold at state $j$ no matter what happens in the future.

**Feasibility Theorems** In the case studies of HIPAA and GLBA, all the policies are or can be rewritten into two general forms. One expresses conditions on performing an action (e.g, §6802 of GLBA); the other expresses future obligation (e.g, §6803 of GLBA).

Based on this observation, we investigate the local feasibility conditions for responsibilities that have the following syntactic structure.

$$\begin{array}{ll} \text{(r1)} & \mathbf{G}\ (\forall \vec{x}.\varphi_c^-(p) \supset \varphi_{past}) \\ \text{(r2)} & \mathbf{G}\ (\forall \vec{x}.\varphi_{past} \supset \varphi_f^+(p)) \\ \text{(r3)} & \mathbf{G}\ (\forall \vec{x}.\varphi_{past} \supset \varphi_f^-(p)) \end{array}$$

We write $\varphi_{past}$ to denote formulas that do not contain future operators. $\varphi_{past}$ captures conditions on the history of the system execution. Responsibilities in (r1) require an agent to not perform actions in the current state unless $\varphi_{past}$ holds. Responsibilities in (r2) require an agent to fulfill obligations if $\varphi_{past}$ holds. Responsibilities in (r3) require an agent to not perform future actions if $\varphi_{past}$ holds. We define the syntactic constructs used in (r1) - (r3) below.

$$\begin{array}{llll}
\text{Conditions} & K^p(\vec{x}) & ::= & \text{contains}(x,q,t) \mid \cdots \\
 & & \mid & K^p(\vec{x}) \wedge K^p(\vec{x}) \mid K^p(\vec{x}) \vee K^p(\vec{x}) \mid \exists y.K^p(\vec{x},y) \\
\text{Actions} & A^p & ::= & \text{send}(p,q,m) \mid \cdots \\
\text{Current Neg Form} & \varphi_c^-(p) & ::= & \bot \mid A^p \mid \varphi_c^-(p) \wedge \varphi \mid \varphi_c^-(p) \vee \varphi_c^-(p) \mid \exists x.\varphi_c^-(p) \\
\text{Future Pos Form} & \varphi_f^+(p) & ::= & \top \mid A^p \mid \varphi_f^+(p) \wedge \varphi_f^+(p) \mid \varphi_f^+(p) \vee \varphi \\
 & & \mid & \exists x.K^p(x) \wedge \varphi_f^+(p) \mid {\downarrow} x.\varphi_f^+(p) \\
 & & \mid & \Diamond\varphi_f^+(p) \mid \Box\varphi_f^+(p) \mid \varphi_f^+(p)\ \mathcal{U}\ \varphi_f^+(p) \\
 & & \mid & \Diamond{\downarrow} x.\, c(x) \wedge \varphi_f^+(p) \mid \varphi_f^+(p)\ \mathcal{U}\ ({\downarrow} x.\, c(x) \wedge \varphi_f^+(p)) \\
\text{Future Neg Form} & \varphi_f^-(p) & ::= & \top \mid \neg A^p \mid \varphi_f^-(p) \wedge \varphi_f^-(p) \mid \varphi_f^-(p) \vee \varphi \\
 & & \mid & \forall x.\varphi_f^-(p) \mid {\downarrow} x.\varphi_f^-(p) \\
 & & \mid & \Diamond\varphi_f^-(p) \mid \Box\varphi_f^-(p) \mid \varphi_f^-(p)\ \mathcal{U}\ \varphi_f^-(p) \\
 & & \mid & \Diamond{\downarrow} x.\, c(x) \wedge \varphi_f^-(p) \mid \varphi_f^-(p)\ \mathcal{U}\ ({\downarrow} x.\, c(x) \wedge \varphi_f^-(p))
\end{array}$$

$\varphi_c^-(p)$ includes formulas that $p$ can cause to be false by planning inactions in the current state. The base case for $\varphi_c^-(p)$ is $A^p$, which denotes an action of which $p$ is a performer (e.g, send$(p,b,m)$). Agent $p$ can cause send$(p,b,m)$ to be false in the current state by not sending $b$ message $m$. $\varphi_f^+(p)$ is similar to $\varphi_c^-(p)$ and it includes all future operators. These are formulas that $p$ can cause to be true by planning actions in future states (e.g, $\Diamond$send$(p,b,m)$). Finally, $\varphi_f^-(p)$ also contains future operators, but the base case is $\neg A^p$. These are formulas that $a$ can cause to be true by planning inactions in future states (e.g, $\Diamond\neg$send$(p,b,m)$).

In the definition of $\varphi_f^+$, existentially quantified variables are guarded by predicates $K^p(\vec{x})$. $K^p(\vec{x})$ are formulas that $p$ can provide a substitution $\delta$ for $\vec{x}$ such that $\delta(K^p(\vec{x}))$ is true and supported by $p$'s knowledge, e.g, contains$(m,\cdots)$.

Finally, variables bound by the freeze operator are guarded by an inequality constraint on the time points $x$ ($c(x)$). We use them to rule out nonsensical formula such as ${\downarrow} x.\Diamond{\downarrow} y.(y < x) \wedge \varphi$ ($y$ should only refer to time points no earlier than $x$ because $y$ shows up under a future operator).

We decide not to include all forms of quantification in $\varphi_c^-$, $\varphi_f^+$ and $\varphi_f^-$. For (r1), it makes little sense to have a universal quantification to the left of the implication, which means that if all instances of some action occur on the trace, then some condition has to hold. Our case studies also supports this decision. The universal quantifiers in $\varphi_f^+(p)$, can be moved to the top-level in (r2). Finally, $\varphi_f^-$ talks about inactions, so it doesn't make sense to have an obligation that requires an agent to selectively not perform an action.

We build up our theorems about feasibility from a single responsibility to composition of responsibilities for a single agent to the composition of responsibilities for a group of agents. The proofs are constructive: they provide concrete plans for agents to discharge their responsibilities. (Details of proofs and auxiliary judgments used in theorems are in Appendix D).

**Theorem 4.1** (Feasibility of one responsibility)**.**

$(r1)$ $\mathbf{G}\ (\forall\vec{x}.\varphi_c^-(p) \supset \varphi_{past})$ *is feasible for agent* $p$

$(r2)$ $\mathbf{G}\ (\forall\vec{x}.\varphi_{past} \supset \varphi_f^+(p))$ *is feasible for agent* $p$
*if* $\vdash \varphi_{past}\ \mathsf{fin}, \cdot; \cdot \vdash \varphi_f^+(p)\ \mathsf{sat}$ *and* $\mathcal{A}_f(\varphi_f^+(p)) \vdash \varphi_{past}$

$(r3)$ $\mathbf{G}\ (\forall\vec{x}.\varphi_{past} \supset \varphi_f^-(p))$ *is feasible for* $p$ *if* $\cdot; \cdot \vdash \varphi_f^+(p)\ \mathsf{sat}$

(r1) is trivially feasible by planning inactions in all states; (r2) is feasible by planning only actions required by $\varphi_f^+$ when $\varphi_{past}$ is true; and (r3) is feasible by planning inactions required by $\varphi_f^-$ in all states. The conditions $\varphi_{past}$, only depend on the history of the trace, so at each state $p$ can decide whether or not these conditions hold; thus, no future prediction is required.

There are several subtle conditions in (r2). Judgment $\vdash \varphi_{past}\ \mathsf{fin}$ ensures that for any given trace, there exists only finite number of substitutions $\delta$ for $\mathrm{fv}(\varphi_{past})$ such that $\delta\varphi_{past}$ holds. This condition allows $p$ to plan only a finite number of actions in each state. Judgment $\mathcal{A}_f(\varphi_f^+(p)) \vdash \varphi_{past}$ holds when actions required by $\varphi_f^+(p)$ do not overlap with actions that $\varphi_{past}$ depends on. We need to make sure that the actions $p$ has planned will not cause more $\varphi_{past}$ to be true; in which case, it is not obvious whether $p$ only needs to perform a finite number of actions. For instance $\varphi_r = \forall m.\ \mathrm{send}(p, q, m) \supset \mathrm{send}(p, q, (m, m))$ is not feasible for $p$ because $p$ needs to perform an infinite number of send actions although it fits the syntactic form presented in (r2). Finally, the judgment $\Sigma; \Gamma \vdash \varphi\ \mathsf{sat}$ checks whether all the time points introduced by the freeze operators in $\varphi_f^+(p)$ are sensible. For instance, nonsensical formulas such as $\downarrow x.\Diamond\downarrow y.(y < x) \land \varphi$ are ruled out. Note that we actually cannot decide purely syntactically if $c(x)$ is satisfiable or not; rules for $\Sigma; \Gamma \vdash \varphi\ \mathsf{sat}$ call a theorem prover to check satisfiability of conditions.

In order to compose plans of different agents, we need to make sure that an agent $p$'s plan is not affected by changes in the current state caused by another agent $b$. Otherwise, the agents would not be able to achieve a stable state without computing a global fixed point across the entire group or imposing a global ordering of agents' actions. We define a syntactic check on a past formula, written $p \vdash \varphi\ \mathsf{StrictPast}$, to ensure that all the current actions that may cause $\varphi$ to be true are completely controlled by $p$. Theorem 4.2 states that a group of agents has a strategy to fulfill a set of responsibilities if each of them can fulfill their own responsibilities, and all the past formulas are not be affected by current actions by other agents.

**Theorem 4.2** (Feasibility composition for multi-agents)**.** *Given a group of agents* $Sa$*, let* $\Phi_p$ *be the set of responsibilities for* $p \in Sa$*. Assume that* $\Phi_p$ *is feasible for* $p$*. If for each* $\varphi_i \in \Phi_p$*, one of the following conditions holds, then the union of* $\Phi_p$ *for all* $p \in Sa$ *is feasible for* $Sa$*.*

1. $\varphi_i = \mathbf{G}\forall\vec{x}.\varphi_c^-(p) \supset \varphi_{past}$*, and* $p \vdash \varphi_{past}\ \mathsf{StrictPast}$
2. $\varphi_i = \mathbf{G}\forall\vec{x}.\varphi_{past} \supset \varphi_f^+(p)$*, and* $p \vdash \varphi_{past}\ \mathsf{StrictPast}$
3. $\varphi_i = \mathbf{G}\forall\vec{x}.\varphi_{past} \supset \varphi_f^-(p)$*, and* $p \vdash \varphi_{past}\ \mathsf{StrictPast}$

So far, we have assumed that an agent can observe all actions in the history, but this is not true in general. An agent $a$ may only view certain part of the state.

**Def.** We say that $\varphi$ is visible to an agent $a$ if all the atomic predicates in $\varphi$ describe states visible to $a$.

**Theorem 4.3.** *If* $\varphi$ *is visible to* $Sa$ *then* $\varphi$ *is local to* $Sa$*.*

Theorem 4.3 and Theorem 4.2 together give us conditions for locally feasible responsibilities for a set of agents.

**Discussion** We have made an assumption that the existence of states is visible to all agents, even though certain actions in those states are not. In other words, all agents are synchronized in lock-step. This assumption simplifies the definition of the merge of two traces since we do not need to consider inserting a state from one trace between two adjacent states on the other. A more general model would allow agents to be completely asynchronous, and the merge of two traces to be defined based on the total ordering of time points. The feasibility theorems given here allow for more responsibilities to be feasible than those that would be allowed in the more general case. For instance, $\diamondminus\varphi \supset \mathrm{send}(p, q, m)$ would not be locally feasible to $p$ in the general case, because $p$ has no way of satisfying this responsibility in states that are not visible to $p$. We plan to investigate the classification of locally feasible responsibilities in the asynchronous case in future work.

# 5 Privacy Policy Enforcement

We present two logic-based methods for enforcing privacy policies. Our first method answers the question: Does a given organizational process respect a given privacy policy? This method is based on a sound *proof system* for LFP and is described in Section 5.1. Although the proof system is obtained by adapting previous proof systems for an intuitionistic logic with fixed-points, $\mu$LJ [8, 17], to our classical logic LFP, we believe that its soundness with respect to trace semantics is a new result. Our second enforcement method audits logs of organizational activity for violations of principals' assigned responsibilities. It is based in a novel tableau-based *model checking* procedure for LFP that we present and prove sound in Section 5.2. Although we develop both methods for LFP, due to the embedding in Section 2.3, both methods apply to PrivacyLFP, as illustrated by examples here.

## 5.1 Auditing Organizational Processes

We present a proof-theoretic method to check whether a given organizational process respects a given privacy policy. We assume that the organizational process is specified in terms of responsibilities $\varphi_{r1}, \ldots, \varphi_{rn}$ of the organization's principals. The privacy policy $\varphi_p$ is also assumed to be specified in LFP. It may, for example, be the formalization of a privacy law such as GLBA or HIPAA from Section 3. Technically, the problem is that of establishing the entailment $(\varphi_{ctx} \wedge \bigwedge_{i=1}^{n} \varphi_{ri}) \supset \varphi_p$, where $\varphi_{ctx}$ relates privacy relevant actions in the organizational process to their counterparts in the policy (examples of such formulas appear in the example at the end of this section). We propose the use of a proof system for LFP to check such entailments. We show that the proof system is sound with respect to the semantics of LFP, which, together with the above entailment, ensures that if principals fulfill their respective responsibilities then the privacy policy is not violated.

Our proof system, presented in the sequence calculus style of Gentzen [22], establishes hypothetical judgments or *sequents* $\Sigma; \Gamma \vdash \Delta$, where $\Gamma$ and $\Delta$ are sets of LFP formulae and $\Sigma$ is a set of variables that occur free in them. The intuitive meaning of $\Sigma; \Gamma \vdash \Delta$ is that for all substitutions $\theta$ for variables in $\Sigma$, $\bigwedge \Gamma\theta$ entails $\bigvee \Gamma\theta$. The rules of inference are shown in Figure 3. Rules for connectives of first-order logic are standard. The interesting rules are those for fixed-point operators, all of which are adapted from similar calculi for the intuitionistic fixed-point logic $\mu$LJ by Baelde [8] and Clairambault [17]. In the rules $\mu$L and $\nu$R, $\psi$ is an arbitrary formula. The rules $\mu$R and $\nu$L simply unfold the fixed-points. The rules $\mu$L and $\nu$R encode induction and co-induction principles for the least and the greatest fixed point operators respectively. We refer the reader to prior work [8, 17] for an explanation of induction and co-induction, but illustrate the use of the rules for the greatest-fixed point operator in proving compliance of organizational processes with privacy policies through an example.

**Theorem 5.1** (Soundness of the sequent calculus)**.** *If $\Sigma; \Gamma \vdash \Delta$ then for all traces $\sigma$ and for all substitutions $\theta$ with $\mathrm{dom}(\theta) = \Sigma$, $(\theta; \sigma \models \bigwedge \Gamma)$ implies $(\theta; \sigma \models \bigvee \Delta)$.*

*Proof.* See Appendix B, Lemma B.1. □

$$\frac{}{\Sigma;\Gamma,\varphi\vdash\varphi,\Delta}\ \text{INIT}\qquad \frac{}{\Sigma;\Gamma\vdash\top,\Delta}\ \top\text{R}\qquad \frac{}{\Sigma;\Gamma,\bot\vdash\Delta}\ \bot\text{L}\qquad \frac{\Sigma;\Gamma\vdash\varphi_1,\Delta \qquad \Sigma;\Gamma\vdash\varphi_2,\Delta}{\Sigma;\Gamma\vdash\varphi_1\wedge\varphi_2,\Delta}\ \wedge\text{R}$$

$$\frac{\Sigma;\Gamma,\varphi_1,\varphi_2\vdash\Delta}{\Sigma;\Gamma,\varphi_1\wedge\varphi_2\vdash\Delta}\ \wedge\text{L}\qquad \frac{\Sigma;\Gamma\vdash\varphi_i,\Delta}{\Sigma;\Gamma\vdash\varphi_1\vee\varphi_2,\Delta}\ \vee\text{R}\qquad \frac{\Sigma;\Gamma,\varphi_1\vdash\Delta \qquad \Sigma;\Gamma,\varphi_2\vdash\Delta}{\Sigma;\Gamma,\varphi_1\vee\varphi_2\vdash\Delta}\ \vee\text{L}\qquad \frac{\Sigma;\Gamma,\varphi\vdash\Delta}{\Sigma;\Gamma\vdash\neg\varphi,\Delta}\ \neg\text{R}$$

$$\frac{\Sigma;\Gamma\vdash\varphi,\Delta}{\Sigma;\Gamma,\neg\varphi\vdash\Delta}\ \neg\text{L}\qquad \frac{\Sigma,a;\Gamma\vdash\varphi\{a/x\},\Delta}{\Sigma;\Gamma\vdash\forall x.\varphi,\Delta}\ \forall\text{R}\qquad \frac{\Sigma;\Gamma,\varphi\{t/x\}\vdash\Delta}{\Sigma;\Gamma,\forall x.\varphi\vdash\Delta}\ \forall\text{L}\qquad \frac{\Sigma;\Gamma\vdash\varphi\{t/x\},\Delta}{\Sigma;\Gamma\vdash\exists x.\varphi,\Delta}\ \exists\text{R}$$

$$\frac{\Sigma,a;\Gamma,\varphi\{a/x\}\vdash\Delta}{\Sigma;\Gamma,\exists x.\varphi\vdash\Delta}\ \exists\text{L}\qquad \frac{\Sigma;\Gamma\vdash\varphi\{\mu X,\vec{x}.\varphi/X\}\{\vec{t}/\vec{x}\},\Delta}{\Sigma;\Gamma\vdash(\mu X(\vec{x}).\varphi)(\vec{t}),\Delta}\ \mu\text{R}$$

$$\frac{\Sigma;\Gamma,\psi\{\vec{t}/\vec{x}\}\vdash\Delta \qquad \Sigma,\vec{y};\Gamma,\varphi\{\lambda\vec{x}.\psi/X\}\{\vec{y}/\vec{x}\}\vdash\psi\{\vec{y}/\vec{x}\}}{\Sigma;\Gamma,(\mu X(\vec{x}).\varphi)(\vec{t})\vdash\Delta}\ \mu\text{L}$$

$$\frac{\Sigma;\Gamma\vdash\psi\{\vec{t}/\vec{x}\},\Delta \qquad \Sigma,\vec{y};\Gamma,\psi\{\vec{y}/\vec{x}\}\vdash\varphi\{\lambda\vec{x}.\psi/X\}\{\vec{y}/\vec{x}\}}{\Sigma;\Gamma\vdash(\nu X(\vec{x}).\varphi)(\vec{t}),\Delta}\ \nu\text{R}\qquad \frac{\Sigma;\Gamma,\varphi\{\nu X,\vec{x}.\varphi/X\}\{\vec{t}/\vec{x}\}\vdash\Delta}{\Sigma;\Gamma,(\nu X(\vec{x}).\varphi)(\vec{t})\vdash\Delta}\ \nu\text{L}$$

Figure 3: Sequent calculus for LFP

Continuing the example from Section 4.1, we would like to audit the processes shown in Figure 2. First, we need to formalize connections between actions of internal processes with their counterparts in higher-level responsibilities – we need to establish that whenever $P$ sends a disclosure, it comes from the disclosure department $D$ and vice versa. This is encoded in the formulas $\varphi_{ctx1}$ and $\varphi_{ctx2}$ below. (Similar requirements for the other two departments as well, but this suffices for our illustration.)

$$\begin{aligned}\varphi_{ctx1} &= \mathbf{G}\ \forall q,t,p'.\, \text{hlsend}(p,q,f\_dis(p,p',q,t)) \supset \\ &\quad\ \text{hlsend}(D,q,f\_dis(p,p',q,t))\\ \varphi_{ctx2} &= \mathbf{G}\ \forall q,t,p'.\, \text{hlsend}(D,q,f\_dis(p,p',q,t)) \supset \\ &\quad\ \text{hlsend}(p,q,f\_dis(p,p',q,t))\end{aligned}$$

Auditing the processes involves discharging the following two proof obligations, where $\varphi^P_{ri}$ $(i = 1, 2)$ is the formula $\varphi_{ri}$ from Figure 2 without the outermost quantification of $p_1$, and with $P$ substituted for $p_1$: (1) $\varphi_{r1}, \varphi_{r2} \vdash \varphi_{pol}$, and (2) $\bigwedge\varphi_{ci}, \varphi_D, \bigwedge\varphi_{SRi}, \bigwedge\varphi_{QRi} \vdash \varphi^p_{r1} \wedge \varphi^p_{r2}$. We illustrate our method by explaining briefly a proof of (1). The proof relies on the co-induction principle for greatest-fixed points codified in the rule $\nu$R. A skeleton of the proof is shown in Figure 4. To apply the rule $\nu$R, we need an appropriate predicate $\psi$ that validates the premises of the rule. Here, an appropriate $\psi$ is the inner body of $\varphi_{r1}$ and $\varphi_{r2}$.

## 5.2 Auditing Responsibilities

Our second enforcement method is an auditing technique which checks logs of organizational activity for violations of principals' assigned responsibilities. Formally, the problem is one of ensuring that a trace $\sigma$ (concretely represented as a log of past activity of principals) satisfies each responsibility, i.e, $\bullet;\sigma \models \varphi_{ri}$ for each responsibility $\varphi_{ri}$.[1] Technically, this is a model checking problem, so, in this section, we develop a local model checking method for LFP and prove it sound.[2] To the best of our knowledge, this is the first local model checking procedure for LFP. Our method builds on prior work on local model checking for the modal $\mu$-calculus [24, 30, 34], which is the propositional fragment of LFP. The modal $\mu$-calculus is interpreted over

[1] $\bullet$ is our notation for an empty set or an empty substitution. Throughout this section, we work only in formulas without free first-order variables.

[2] A model checking method is called "local" if it does not explicitly compute the entire interpretation of each recursively defined predicate.

$$
\begin{aligned}
\psi(p_1, p_2, m) = \forall q, t.\, &\mathrm{contains}(m, q, t) \supset \\
&(\diamondminus \mathrm{hlsend}(p_1, q, f\_dis(p_1, p_2, q, t)) \wedge \forall p', m'.\mathrm{hlsend}(p', p_1, m') \wedge \\
&\ \mathrm{contains}(m', q, t) \supset \diamondminus \mathrm{send}(p', p_1, f\_reply\_maysend(p', p_2, m')))
\end{aligned}
$$

($\varphi_b$ is the body of maysend)

$$
\dfrac{\dfrac{\cdots \qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\psi\{p', p_2, m''/p_1, p_2, m\} \vdash \psi\{p', p_2, m''/p_1, p_2, m\}}\ \textsc{init}}{\cdots}}{\begin{array}{l}\varphi_{r2}, \mathrm{send}(p', p_1, f\_reply\_maysend(p', p_2, m'')) \\ \vdash \psi\{p', p_2, m''/p_1, p_2, m\}\end{array}}}{\cdots}}{\begin{array}{l}\psi \vdash \forall q, t.\, \mathrm{contains}(m, q, t) \supset \\ \quad (\diamondminus \mathrm{hlsend}(p_1, q, f\_dis(p_1, p_2, q, t) \wedge \\ \quad\quad \forall p', m''.\mathrm{hlsend}(p', p_1, m'') \\ \quad\quad\quad \wedge\, \mathrm{contains}(m'', q, t) \supset \\ \quad\quad\quad \diamondminus \psi\{p', p_2, m''/p_1, p_2, m\}))\end{array}}}{\begin{array}{l}\varphi_{r2}, \psi\{p_1', p_2', m'/p_1, p_2, m\} \\ \vdash (\nu \mathrm{maysend}(p_1, p_2, m).\varphi_b)(p_1', p_2', m')\end{array}}\ \nu\mathrm{R}}{\dfrac{\cdots \{\forall R, \forall L, \vee R, \neg R, \vee L, \neg L\}}{\varphi_{r1}, \varphi_{r2} \vdash \varphi_{pol}}}
$$

Figure 4: Example Proof Tree

Kripke structures and the objective of model checking for it is to find for each recursively defined predicate, the set of worlds of the Kripke structure in which the predicate is true. The key insight in generalizing model checking from the modal $\mu$-calculus to LFP is to view each tuple of terms as a world and to relate satisfaction relations in the modal $\mu$-calculus and LFP by saying that the world $\vec{t}$ satisfies the LFP predicate $P$ if and only if $P(\vec{t})$ holds. Given this insight, our model checking method is an unsurprising generalization of Winskel's method [34] for model-checking the modal $\mu$-calculus.

We formalize the model-checking procedure as semantic tableaus. We work only in NNF formulas (we showed in Section 2.1 that every LFP formula can be translated to NNF through DeMorgan's laws). To deal with greatest fixed-points in tableaus, we rely on equations, which have the form $X \Rightarrow \lambda\vec{x}.\ \varphi$ ($\varphi$ may mention both $X$ and $\vec{x}$). Given this equation, the interpretation of $X$ is the largest relation that equates the two sides of the definition $X(\vec{x}) \triangleq \varphi$ semantically. We call $X$ the defined variable of the equation and $\vec{x}$ the equation's parameters and $\varphi$ its body. A list of equations $\mathcal{E}$ is a list $E_1, \ldots, E_n$ with the constraints that no predicate variable be defined twice in the list, and for each $i$, the body of $E_i$ may not mention predicate variables defined in $E_{i+1}, \ldots, E_n$.

Our semantic tableaus work with formulas without first-order variables and infer judgments of the form $\sigma; \mathcal{E}; \Delta \vdash \varphi$, where $\Delta$ is a set of pairs of the form $X : S$, which intuitively means that $X\ \vec{t}$ holds for each $\vec{t} \in S$ and $\mathcal{E}$ is a list of equations that defines all predicate variables free in $\Delta$ and $\varphi$. ($S$ is a finite set of tuples.) $\sigma$ is an interpretation of all predicate symbols $P$ in $\mathcal{E}$, $\Delta$, and $\varphi$ given to us as a trace against which we are auditing. Roughly, the meaning of the entire judgment is that $\varphi$ is true in the interpretation $\sigma$ for predicate symbols and the largest possible interpretation for each equation that also includes $X\ \vec{t}$ for every $X : S \in \Delta$ and $\vec{t} \in S$.

To check that $\bullet; \sigma \models \varphi$, the tableau procedure starts with the judgment $\sigma; \bullet; \bullet \vdash \varphi$ and tries to construct a derivation by applying the rules of Figure 5 backwards. A branch closes or successfully ends when it matches a rule whose premises are satisfiable and do not contain the symbol $\vdash$. Most of the rules of Figure 5 are straightforward and correspond to the semantics of LFP. The interesting rules are those for fixed points. The rule for $(\mu X(\vec{x}).\varphi)(\vec{t})$ *unrolls* the fixed-point. The soundness of this rule is a consequence of the semantics of

$$\frac{[\![\vec{t}\,]\!]^{\bullet} \in \mathcal{I}_{\sigma}(P)}{\sigma;\mathcal{E};\Delta \vdash P\ \vec{t}}\ \mathrm{INIT} \qquad \frac{[\![\vec{t}\,]\!]^{\bullet} \notin \mathcal{I}_{\sigma}(P)}{\sigma;\mathcal{E};\Delta \vdash \neg(P\ \vec{t})}\ \neg \qquad \frac{}{\sigma;\mathcal{E};\Delta \vdash \top}\ \top \qquad \frac{\sigma;\mathcal{E};\Delta \vdash \varphi_1 \qquad \sigma;\mathcal{E};\Delta \vdash \varphi_2}{\sigma;\mathcal{E};\Delta \vdash \varphi_1 \wedge \varphi_2}\ \wedge$$

$$\frac{\sigma;\mathcal{E};\Delta \vdash \varphi_1}{\sigma;\mathcal{E};\Delta \vdash \varphi_1 \vee \varphi_2}\ \vee 1 \qquad \frac{\sigma;\mathcal{E};\Delta \vdash \varphi_2}{\sigma;\mathcal{E};\Delta \vdash \varphi_1 \vee \varphi_2}\ \vee 2 \qquad \frac{\text{all } d \in \mathrm{D}.\ (\sigma;\mathcal{E};\Delta \vdash \varphi\{d/x\})}{\sigma;\mathcal{E};\Delta \vdash \forall x.\varphi}\ \forall$$

$$\frac{d \in \mathrm{D} \qquad \sigma;\mathcal{E};\Delta \vdash \varphi\{d/x\}}{\sigma;\mathcal{E};\Delta \vdash \exists x.\varphi}\ \exists \qquad \frac{\sigma;\mathcal{E};\Delta \vdash \varphi\{\vec{t}/\vec{x}\}\{(\mu X, \vec{x}.\ \varphi)/X\}}{\sigma;\mathcal{E};\Delta \vdash (\mu X(\vec{x}).\varphi)(\vec{t})}\ \mu$$

$$\frac{\sigma;\mathcal{E}, X \Rightarrow \lambda\vec{x}.\varphi;\Delta, X : \{\} \vdash X\ \vec{t} \qquad (X \text{ fresh})}{\sigma;\mathcal{E};\Delta \vdash (\nu X(\vec{x}).\varphi)(\vec{t})}\ \nu \qquad \frac{[\![\vec{t}\,]\!]^{\bullet} \in [\![S]\!]^{\bullet}}{\sigma;\mathcal{E};\Delta, X : S \vdash X\ \vec{t}}\ X1$$

$$\frac{[\![\vec{t}\,]\!]^{\bullet} \notin [\![S]\!]^{\bullet} \qquad (X \Rightarrow \lambda\vec{x}.\varphi) \in \mathcal{E} \qquad \sigma;\mathcal{E};\Delta, X : S \cup \{\vec{t}\} \vdash \varphi\{\vec{t}/\vec{x}\}}{\sigma;\mathcal{E};\Delta, X : S \vdash X\ \vec{t}}\ X2$$

Figure 5: Semantic tableau for model-checking LFP. The rules are applied backwards.

LFP. The rule for $(\nu X(\vec{x}).\varphi)(\vec{t})$ creates a fresh predicate name $X$ for the predicate defined by the fixed-point and stores its definition as the equation $X \Rightarrow \lambda\vec{x}.\varphi$ in $\mathcal{E}$. The equation is looked up in the rule $X2$ for checking $X\ \vec{t}$. In the third premise of that rule, the fact that $X\ \vec{t}$ has been encountered on the branch is recorded in $\Delta$. This ensures that if $X\ \vec{t}$ is encountered again, then the branch closes (rule $X1$). Winskel [34] proved that admitting cycles for greatest-fixed points in this manner is sound in the propositional case. Our soundness theorem (Theorem 5.2) extends that result to the first-order case.

**Theorem 5.2** (Soundness of tableau)**.** *If $\sigma;\bullet;\bullet \vdash \varphi$ has a successful tableau, then $\bullet;\sigma \models \varphi$.*

*Proof.* See Appendix C, Theorem C.5. □

An important practical consideration in any model checking procedure for first-order logic is treatment of quantifiers. The rules for quantifiers in Figure 5 require guessing a correct substitution for an existentially bound variable and iterating over all elements of the domain of interpretation, D, for a universally bound variable, both of which may be impossible if D is infinite. In practice, the problem can be addressed by assuming that all quantifiers are guarded by formulas that restrict the relevant substitutions for the bound variables to finite sets, as in guarded first-order logic [5].

We illustrate the use of model-checking in auditing traces for violations of privacy-related responsibilities from the Example of Figure 2, which does not include fixed-points. Readers should bear in mind that even though this example does not use fixed-points, their treatment, especially that of greatest fixed-points, is the technically challenging part of the method.

Suppose that $E$ and $Q$ are principals (whose details are irrelevant to this example) and $M$ is a message which contains some information about attribute $T$ of $Q$. Consider a trace $\sigma$ with just two states $\sigma = s_0 s_1$, and each of the two states containing exactly one message transmission as follows:

$$\begin{array}{lll} s_0 & D \to Q & : \ f\_dis(P, E, Q, T) \\ s_1 & D \to SR & : \ f\_sent\_dis(E, Q, T) \end{array}$$

In state $s_0$ the disclosure department $D$ informs principal $Q$ that its information about attribute $T$ may be sent to $E$ by $P$. In the next state $s_1$, $D$ informs $SR$ about this disclosure. We are interested in checking whether $D$ has violated its responsibility $\varphi_D$ from Figure 2. In this example it hasn't. To check that this is the case, we must construct a tableau for $\sigma;\bullet;\bullet \vdash \mathbf{G}\ \forall p', m, q, t.\,(\mathrm{send}(D, SR, f\_sent\_dis(p', q, t)) \supset \diamondminus \mathrm{hlsend}(D, q, f\_dis(P, p', q, t)))$. Expanding the syntax into LFP through the embedding of Section 2.3, we must check that $\sigma; \bullet; \bullet \vdash \forall s, p', m, q, t. (\neg \mathrm{send}(s, D, SR, f\_sent\_dis(p', q, t))) \vee \exists s'. ((s' \leq_{\mathsf{st}} s) \wedge \mathrm{hlsend}(s', D, q, f\_dis(P, p', q, t)))$. Using rule $\forall$ from Figure 5, we must check for every $s$, $p'$, $m$, $q$, $t$ that $\sigma; \bullet; \bullet \vdash (\neg \mathrm{send}(s, D, SR, f\_sent\_dis(p', q, t))) \vee \exists s'. ((s' \leq_{\mathsf{st}} s) \wedge \mathrm{hlsend}(s', P, q, f\_dis(P, p', q, t)))$. For $(s, p', m, q, t) \neq (s_1, E, M, Q, T)$, the branch $\neg \mathrm{send}(s, D, SR, f\_sent\_dis(p', q, t))$ succeeds by $\vee 1$ and INIT. Hence, it only remains to check that $\sigma; \bullet; \bullet \vdash \exists s'. ((s' \leq_{\mathsf{st}} s_1) \wedge \mathrm{hlsend}(s', D, Q, f\_dis(P, E, Q, T)))$. Choosing $s' = s_0$ in rule $\exists$, this reduces to $\sigma; \bullet; \bullet \vdash (s_0 \leq_{\mathsf{st}} s_1) \wedge \mathrm{hlsend}(s_0, D, Q, f\_dis(P, E, Q, T))$. The conjunct $s_0 \leq_{\mathsf{st}} s_1$ succeeds by definition of the trace, so we must show only that $\sigma; \bullet; \bullet \vdash \mathrm{hlsend}(s_0, D, Q, f\_dis(P, E, Q, T))$, which succeeds immediately by rule INIT.

# 6 Related Work

The core logic used for the technical work in this paper is the least-fixed point logic (LFP) [13, 28]. However, the proof-theory for fixed-points presented in Section 5.1 is based on an unrelated source – the *intuitionistic* logic $\mu$LJ – that has been used in the past as a logical framework for specifying and reasoning about formal systems [8, 17]. Besides adapting that work to our classical setting, we also prove the proof-system sound with respect to trace semantics. Our model-checking method (Section 5.2) is a first-order extension of prior work on model-checking the *propositional* modal $\mu$-calculus [24, 30, 34], most notably the work of Winskel [34].

Privacy languages such as EPAL [6, 7] and XACML [4] are formulated as access control frameworks. EPAL and XACML do not possess first-class temporal modalities, but have a much weaker uninterpreted obligation symbol for representing future requirements. Our logic has its rich temporal and obligation constructs and is, therefore, more expressive than EPAL and XACML. P3P [1, 15, 29] is a privacy language targeted exclusively to web sites, but due to its domain-specific design it is unsuited for expressing privacy policies based on laws like HIPAA and GLBA. RBAC languages focus on access control [18, 23, 25] but lack a notion of data attribute as well as temporal modalities needed to express privacy policies.

Choosing deontic logic, rather than temporal logic, as a foundation, Dinesh *et al.* have developed a logic for reasoning about conditions and exceptions in privacy laws [21]. The approach of Dinesh *et al.* is advantageous in that it simplifies the task of formalizing the law clause by clause: there is no need to modify previously formalized clauses if exceptions appear in later paragraphs. Further investigation is needed to determine whether their ideas can be adapted to our logic.

# 7 Conclusion

We presented the logic PrivacyLFP and used it to express role-based responsibilities of agents in organizational processes. We presented a semantic locality criterion to characterize "reasonable" responsibilities that agents (or groups of agents) have a strategy to discharge, and easily checkable, sound syntactic characterizations of responsibilities that meet this criterion. We develop policy enforcement techniques based on a sound proof system and an auditing procedure for PrivacyLFP based on a tableau-based model checking algorithm we develop. We illustrated these enforcement techniques using a representative example of an organizational process. In future work, we plan to apply these techniques to larger organizational processes, formalize other privacy regulations, and develop tool support for policy enforcement.

**Acknowledgments** This work was partially supported by the U.S. Army Research Office contract on Perpetually Available and Secure Information Systems (DAAD19-02-1-0389) to Carnegie Mellon CyLab, the NSF Science and Technology Center TRUST, the NSF CyberTrust grant "Privacy, Compliance and Information Risk in Complex Organizational Processes" and the AFOSR MURI "Collaborative Policies and Assured Information Sharing". The first author was also supported under an NSF Graduate Research Fellowship. Any opinions, findings, conclusions, or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the U.S. Army Research Office, the National Science Foundation, or the Air Force Office of Scientific Research.

The authors thank Arunesh Sinha and Nataliia Bielova for discussions leading to some of the ideas presented in Section 4 and David Baelde for technical discussions on proof rules for fixed-points.

# References


[1] M. S. Ackerman, L. F. Cranor, and J. Reagle. Privacy in e-commerce: Examing user scenarios and privacy preferences. In *Proceedings of the 1st ACM Conference on Electronic Commerce*, pages 1–8, 1999.

[2] R. Alur and T. A. Henzinger. A really temporal logic. *Journal of the ACM*, 41(1):181–203, January 1994.

[3] R. Alur, T. A. Henzinger, and O. Kupferman. Alternating-time temporal logic. *Journal of the ACM*, 49(5):672–713, 2002.

[4] A. Anderson, A. Nadalin, B. Parducci, D. Engovatov, E. Coyne, F. Siebenlist, H. Lockhart, M. McIntosh, M. Kudo, P. Humenn, R. Jacobson, S. Proctor, S. Godik, S. Anderson, and T. Moses. Extensible Access Control Markup Language (XACML) version 2.0, 2004.

[5] H. Andréka, I. Németi, and J. van Benthem. Modal languages and bounded fragments of predicate logic. *Journal of Philosophical Logic*, 27(3):217–274, 1998.

[6] M. Backes, G. Karjoth, W. Bagga, and M. Schunter. Efficient comparsion of enterprise privacy policies. In *Proceedings of the 2004 ACM Symposium on Applied Computing*, pages 375–382, 2004.

[7] M. Backes, B. Pfitzmann, and M. Schunter. A toolkit for managing enterprise privacy policies. In *European Symposium on Research in Computer Security*, volume 2808 of *LNCS*, pages 101–119, 2003.

[8] D. Baelde. *A linear approach to the proof-theory of least and greatest fixed points*. PhD thesis, École Polytechnique, 2009.

[9] A. Barth. *Design and Analysis of Privacy Policies*. Phd thesis, Stanford University, 2008.

[10] A. Barth, A. Datta, J. C. Mitchell, and H. Nissenbaum. Privacy and contextual integrity: Framework and applications. In *Proceedings of the 27th IEEE Symposium on Security and Privacy*, pages 184–198, May 2006.

[11] A. Barth, A. Datta, J. C. Mitchell, and S. Sundaram. Privacy and utility in business processes. In *Proceedings of the 20th IEEE Computer Security Foundations Symposium*, pages 279–294, July 2007.

[12] P. Blackburn. Representation, reasoning, and relational structures: A hybrid logic manifesto. *Logic Journal of IGPL*, 8(3):339–365, 2000.

[13] J. Bradfield and C. Stirling. *The Handbook of Modal Logic*, chapter Modal Mu-Calculi, pages 721–756. Elsevier, 2006.

[14] T. Braüner and V. de Paiva. Towards constructive hybrid logic. In *Electronic Proceedings of Methods for Modalities 3 (M4M3)*, 2003. Online at http://m4m.loria.fr/M4M3/Papers/brauner.ps.gz.

[15] S. Byers, L. F. Cranor, and D. Kormann. Automated analysis of P3P-enabled web sites. In *Proceedings of the 5th International Conference on Electronic Commerce*, pages 326–338, 2003.

[16] R. Chadha, D. Macedonio, and V. Sassone. A hybrid intuitionistic logic: Semantics and decidability. *Journal of Logic and Computation*, 16:27–59(33), 2006.

[17] P. Clairambault. *Logique et interaction: Une étude sémantique de la totalité*. PhD thesis, Université Paris Diderot - Paris 7, 2010.

[18] J. Crampton. On permissions, inheritance, and role hierarchies. In *Proceedings of the 10th ACM Conference on Computer and Communication Security*, pages 85–92, 2003.

[19] Deloitte and Ponemnon Institute. Privacy and Data Protection Survey. White Paper, December 2007.

[20] H. DeYoung, D. Garg, D. Kaynar, and A. Datta. Logical specification of the GLBA and HIPAA privacy regulations. Technical Report CMU-CyLab-10-007, Carnegie Mellon University, 2010.

[21] N. Dinesh, A. K. Joshi, I. Lee, and O. Sokolsky. Reasoning about conditions and exceptions to laws in regulatory conformance checking. In *Proceedings of the Ninth International Conference on Deontic Logic in Computer Science*, volume 5076 of *LNAI*, pages 110–124, July 2008.

[22] G. Gentzen. Untersuchungen über das logische Schließen. *Mathematische Zeitschrift*, 39:176–210, 405–431, 1935. English translation in M. E. Szabo, editor, *The Collected Papers of Gerhard Gentzen*, pages 68–131, North-Holland, 1969.

[23] S. Jajodia, P. Samarati, M. L. Sapino, and V. S. Subrahmanian. Flexible support for multiple access control policies. *ACM Transactions on Database Systems*, 26(2):214–260, 2001.

[24] K. G. Larsen. Proof system for hennessy-milner logic with recursion. In *Proceedings of the 13th Colloquium on Trees in Algebra and Programming (CAAP'88)*, pages 215–230, 1988.

[25] N. Li, J. C. Mitchell, and W. H. Winsborough. Design of a role-based trust management framework. In *Proceedings of the 2002 IEEE Symposium on Security and Privacy*, pages 114–130. IEEE Computer Society Press, May 2002.

[26] Z. Manna and A. Pnueli. *Temporal Verification of Reactive Systems: Safety.* Springer-Verlag, 1995.

[27] H. Nissenbaum. *Privacy in Context: Technology, Policy, and the Integrity of Social Life.* Stanford University Press, 2010.

[28] T. Räsch. Introduction to guarded logics. In *Automata, Logics, and Infinite Games*, volume 2500 of *Lecture Notes in Computer Science*, pages 321–342. Springer-Verlag, 2002.

[29] J. Reagle and L. F. Cranor. The platform for privacy preferences. *Communications of the ACM*, 42(2):48–55, 1999.

[30] C. Stirling and D. Walker. Local model checking in the modal mu-calculus. *Theoretical Computer Science*, 89(1):161–177, 1991.

[31] A. Tarski. A lattice-theoretical fixpoint theorem and its applications. *Pacific Journal of Mathematics*, 5(2):283–309, 1955.

[32] US Congress. Gramm-Leach-Bliley Act, Financial Privacy Rule. 15 USC §6801–§6809, November 1999. Available at http://www.law.cornell.edu/uscode/usc_sup_01_15_10_94_20_I.html.

[33] US Congress. Health Insurance Portability and Accountability Act of 1996, Privacy Rule. 45 CFR 164, August 2002. Available at http://www.access.gpo.gov/nara/cfr/waisidx_07/45cfr164_07.html.

[34] G. Winskel. A note on model checking the modal $\nu$-calculus. *Theoretical Computer Science*, 83(1):157–167, 1991.

# A Basic Theory of LFP

We develop some preliminary theory of LFP's semantics. Our proofs of soundness of proof-theory and model checking (Section 5) rely on this theory.

**Lemma A.1** (Term substitution)**.** *The following hold:*

1. $\theta, \theta'; \mathcal{I} \models \varphi$ *if and only if* $\theta'; \mathcal{I} \models \varphi\theta$
2. $F^{X,\vec{x}}_{\mathcal{I},(\theta,\theta')}(\varphi) = F^{X,\vec{x}}_{\mathcal{I},\theta'}(\varphi\theta)$

*Proof.* First observe that since $[\![\, t \,]\!]^{\theta,\theta'}$ must be homomorphic in the structure of terms, $[\![\, t\theta \,]\!]^{\theta'} = [\![\, t \,]\!]^{\theta,\theta'}$. The proof of both statements then follows by lexicographic induction, first on $\varphi$ and then on (1) < (2). ☐

**Lemma A.2** (Variable substitution)**.** *Let $\varphi$ be a formula, possibly containing the distinguished variables $\vec{x}$, and let $X$ be a predicate variable of arity $|\vec{x}|$. Then,*

1. $\theta; \mathcal{I} \models \psi\{(\lambda\vec{x}.\varphi)/X\}$ *if and only if* $\theta; \mathcal{I}, X \mapsto \{\vec{d} \mid \theta, \vec{x} \mapsto \vec{d}; \mathcal{I} \models \varphi\} \models \psi$.
2. $F^{Y,\vec{y}}_{\mathcal{I},\theta}(\psi\{(\lambda\vec{x}.\varphi)/X\}) = F^{Y,\vec{y}}_{(\mathcal{I},X \mapsto \{\vec{d} \mid \theta,\vec{x} \mapsto \vec{d}; \mathcal{I} \models \varphi\}),\theta'}(\psi)$.

*Proof.* The proof of both statements follows by lexicographic induction, first on $\psi$ and then on (1) < (2). ☐

## A.1 Fixed-point Unrolling

We prove some results about unrolling of fixed-points.

**Lemma A.3** (Unrolling lemma)**.** *The following hold:*

1. $\theta; \mathcal{I} \models (\mu X(\vec{x}).\ \varphi)\ \vec{t}$ *if and only if* $\theta, \vec{x} \mapsto [\![\vec{t}\,]\!]^{\theta}; \mathcal{I} \models \varphi\{(\mu X(\vec{x}).\ \varphi)/X\}$
2. $\theta; \mathcal{I} \models (\nu X(\vec{x}).\ \varphi)\ \vec{t}$ *if and only if* $\theta, \vec{x} \mapsto [\![\vec{t}\,]\!]^{\theta}; \mathcal{I} \models \varphi\{(\nu X(\vec{x}).\ \varphi)/X\}$

*Proof.* We prove (1) below. The proof of (2) is identical except that $\mu$ is replaced by $\nu$ everywhere (this works because the proof below relies *only on* $\mu X(\vec{x}).\ \varphi$ being a fixed-point, not on it being the *least* fixed-point).

**Proof of (1).** We have:

$$
\begin{array}{lll}
 & \theta; \mathcal{I} \models (\mu X(\vec{x}).\ \varphi)\ \vec{t} & \\
\leftrightarrow & [\![\vec{t}\,]\!]^{\theta} \in \mu F^{X,\vec{x}}_{\mathcal{I},\theta}(\varphi) & \text{(Defn.)} \\
\leftrightarrow & [\![\vec{t}\,]\!]^{\theta} \in F^{X,\vec{x}}_{\mathcal{I},\theta}(\varphi)(\mu F^{X,\vec{x}}_{\mathcal{I},\theta}(\varphi)) & (\mu F^{X,\vec{x}}_{\mathcal{I},\theta}(\varphi) \text{ is a fixed-point of } F^{X,\vec{x}}_{\mathcal{I},\theta}(\varphi)) \\
\leftrightarrow & [\![\vec{t}\,]\!]^{\theta} \in \{\vec{d} \mid \theta, \vec{x} \mapsto \vec{d}; \mathcal{I}, X \mapsto \mu F^{X,\vec{x}}_{\mathcal{I},\theta}(\varphi) \models \varphi\} & \text{(Defn.)} \\
\leftrightarrow & \theta, \vec{x} \mapsto [\![\vec{t}\,]\!]^{\theta}; \mathcal{I}, X \mapsto \mu F^{X,\vec{x}}_{\mathcal{I},\theta}(\varphi) \models \varphi & \text{(Set-theory)} \\
\leftrightarrow & \theta, \vec{x} \mapsto [\![\vec{t}\,]\!]^{\theta}; \mathcal{I}, X \mapsto \{[\![\vec{t}\,]\!]^{\theta} \mid \theta; \mathcal{I} \models (\mu X(\vec{x}).\ \varphi)\ \vec{t}\} \models \varphi & \text{(Defn. of } \theta; \mathcal{I} \models (\mu X(\vec{x}).\ \varphi)\ \vec{t}) \\
\leftrightarrow & \theta, \vec{x} \mapsto [\![\vec{t}\,]\!]^{\theta}; \mathcal{I}, X \mapsto \{[\![\vec{t}\,]\!]^{\theta} \mid \theta, \vec{x} \mapsto [\![\vec{t}\,]\!]^{\theta}; \mathcal{I} \models (\mu X(\vec{x}).\ \varphi)\ \vec{x}\} \models \varphi & \text{(Lemma A.1)} \\
\leftrightarrow & \theta, \vec{x} \mapsto [\![\vec{t}\,]\!]^{\theta}; \mathcal{I}, X \mapsto \{\vec{d} \mid \theta, \vec{x} \mapsto \vec{d}; \mathcal{I} \models (\mu X(\vec{x}).\ \varphi)\ \vec{x}\} \models \varphi & \\
\leftrightarrow & \theta, \vec{x} \mapsto [\![\vec{t}\,]\!]^{\theta}; \mathcal{I} \models \varphi\{(\lambda\vec{x}.\ (\mu X(\vec{x}).\ \varphi)\ \vec{x})/X\} & \text{(Lemma A.2)} \\
\leftrightarrow & \theta, \vec{x} \mapsto [\![\vec{t}\,]\!]^{\theta}; \mathcal{I} \models \varphi\{(\mu X(\vec{x}).\ \varphi)/X\} & ((\lambda\vec{x}.\ f\ \vec{x}) = f)
\end{array}
$$

☐

**Finite unrolling.** Next we consider finite unrolling of fixed-points and its semantic interpretation. We define the function $U^{X,\vec{x}}_{\psi,n}(\varphi)$, which unrolls the definition $(X(\vec{x}) = \varphi)$ $n$ times, using $\psi$ for the base case ($\psi$ may mention contain the free variables $\vec{x}$). The result is a predicate of the form $\lambda\vec{x}.\ \varphi'$. The function $U^{X,\vec{x}}_{\psi,n}(\varphi)$ is defined by induction on $n$ as follows.

$$\begin{array}{lcl} U^{X,\vec{x}}_{\psi,0}(\varphi) & = & \lambda\vec{x}.\psi \\ U^{X,\vec{x}}_{\psi,n+1}(\varphi) & = & \lambda\vec{x}.\varphi\{U^{X,\vec{x}}_{\psi,n}(\varphi)/X\} \end{array}$$

**Lemma A.4** (Finite unrolling)**.** *$\theta;\mathcal{I} \models (U^{X,\vec{x}}_{\psi,n}(\varphi))\ \vec{t}$ if and only if $[\![\vec{t}]\!]^\theta \in (F^{X,\vec{x}}_{\mathcal{I},\theta}(\varphi))^n(\{\vec{d} \mid \theta, \vec{x} \mapsto \vec{d}; \mathcal{I} \models \psi\})$.*

*Proof.* By induction on $n$.

**Case.** $n = 0$

$$\begin{array}{lll} & \theta;\mathcal{I} \models (U^{X,\vec{x}}_{\psi,0}(\varphi))\ \vec{t} & \\ \leftrightarrow & \theta;\mathcal{I} \models (\lambda\vec{x}.\psi)\ \vec{t} & (\text{Defn. of } U^{X,\vec{x}}_{\psi,0}(\varphi)) \\ \leftrightarrow & \theta;\mathcal{I} \models \psi\{\vec{t}/\vec{x}\} & \\ \leftrightarrow & \theta, \vec{x} \mapsto [\![\vec{t}]\!]^\theta; \mathcal{I} \models \psi & (\text{Lemma A.1}) \\ \leftrightarrow & [\![\vec{t}]\!]^\theta \in \{\vec{d} \mid \theta, \vec{x} \mapsto \vec{d}; \mathcal{I} \models \psi\} & (\text{Set-theory}) \\ \leftrightarrow & [\![\vec{t}]\!]^\theta \in (F^{X,\vec{x}}_{\mathcal{I},\theta}(\varphi))^0(\{\vec{d} \mid \theta, \vec{x} \mapsto \vec{d}; \mathcal{I} \models \psi\}) & (f^0(x) = x) \end{array}$$

**Case.** $n = k + 1$

$$\begin{array}{lll} & \theta;\mathcal{I} \models (U^{X,\vec{x}}_{\psi,k+1}(\varphi))\ \vec{t} & \\ \leftrightarrow & \theta;\mathcal{I} \models (\lambda\vec{x}.\ \varphi\{U^{X,\vec{x}}_{\psi,k}(\varphi)/X\})\ \vec{t} & (\text{Defn. of } U^{X,\vec{x}}_{\psi,k+1}(\varphi)) \\ \leftrightarrow & \theta;\mathcal{I} \models \varphi\{\vec{t}/\vec{x}\}\{U^{X,\vec{x}}_{\psi,k}(\varphi)/X\} & \\ \leftrightarrow & \theta;\mathcal{I} \models \varphi\{\vec{t}/\vec{x}\}\{(\lambda\vec{x}.\ (U^{X,\vec{x}}_{\psi,k}(\varphi))\ \vec{x})/X\} & (f = \lambda\vec{x}.(f\ \vec{x})) \\ \leftrightarrow & \theta;\mathcal{I}, X \mapsto \{\vec{d} \mid \theta, \vec{x} \mapsto \vec{d}; \mathcal{I} \models U^{X,\vec{x}}_{\psi,k}(\varphi)\ \vec{x}\} \models \varphi\{\vec{t}/\vec{x}\} & (\text{Lemma A.2}) \\ \leftrightarrow & \theta;\mathcal{I}, X \mapsto \{[\![\vec{t}]\!]^\theta \mid \theta;\mathcal{I} \models U^{X,\vec{x}}_{\psi,k}(\varphi)\ \vec{t}\} \models \varphi\{\vec{t}/\vec{x}\} & (\text{Lemma A.1}) \\ \leftrightarrow & \theta, \vec{x} \mapsto [\![\vec{t}]\!]^\theta; \mathcal{I}, X \mapsto \{[\![\vec{t}]\!]^\theta \mid \theta;\mathcal{I} \models U^{X,\vec{x}}_{\psi,k}(\varphi)\ \vec{t}\} \models \varphi & (\text{Lemma A.1}) \\ \leftrightarrow & \theta, \vec{x} \mapsto [\![\vec{t}]\!]^\theta; \mathcal{I}, X \mapsto (F^{X,\vec{x}}_{\mathcal{I},\theta}(\varphi))^k(\{\vec{d} \mid \theta, \vec{x} \mapsto \vec{d}; \mathcal{I} \models \psi\}) \models \varphi & (\text{i.h.}) \\ \leftrightarrow & [\![\vec{t}]\!]^\theta \in (F^{X,\vec{x}}_{\mathcal{I},\theta}(\varphi))^{k+1}(\{\vec{d} \mid \theta, \vec{x} \mapsto \vec{d}; \mathcal{I} \models \psi\}) & (\text{Defn. of } F^{X,\vec{x}}_{\mathcal{I},\theta}(\varphi)) \end{array}$$

□

# B Proof System for LFP

We prove the proof theory of LFP sound with respect to the semantics of LFP.

**Lemma B.1** (Soundness of sequent calculus)**.** *If $\Sigma;\Gamma \vdash \Delta$ then for all interpretation $\mathcal{I}$, for all substitution $\theta$, where $\mathrm{dom}(\theta) = \Sigma$, $\theta;\mathcal{I} \vDash \bigwedge\Gamma$ implies $\theta;\mathcal{I} \vDash \bigvee\Delta$*

*Proof.* By induction on the structure of the derivation $\Sigma;\Gamma \vdash \Delta$.

**Case.** $\mu$R

$$\frac{\mathcal{E} :: \Sigma;\Gamma \vdash \varphi\{\mu X(\vec{x}).\varphi/X\}\{\vec{t}/\vec{x}\}, \Delta}{\Sigma;\Gamma \vdash \mu X(\vec{x}).\varphi\ \vec{t}, \Delta}\ \mu\text{R}$$

By assumptions, given any $\mathcal{I}$, $\theta$ where $\mathrm{dom}(\theta) = \Sigma$

$\theta;\mathcal{I} \vDash \bigwedge\Gamma$

By I.H. on $\mathcal{E}$

$\theta;\mathcal{I} \vDash \varphi\{\mu X(\vec{x}).\varphi/X\}\{\vec{t}/\vec{x}\} \vee \bigvee\Delta$

By Lemma A.3

$\theta;\mathcal{I} \vDash \varphi\{\mu X(\vec{x}).\varphi/X\}\{\vec{t}/\vec{x}\}$ iff $\theta, \vec{x} \mapsto [\![\vec{t}\,]\!]^\theta; \mathcal{I} \vDash \mu X(\vec{x}).\varphi\ \vec{x}$

By Lemma A.1,

$\theta, \vec{x} \mapsto [\![\vec{t}\,]\!]^\theta; \mathcal{I} \vDash \mu X(\vec{x}).\varphi\ \vec{x}$ iff $\theta;\mathcal{I} \vDash \mu X(\vec{x}).\varphi\ \vec{t}$

By the above two,

$\theta;\mathcal{I} \vDash \mu X(\vec{x}).\varphi\ \vec{t} \vee \bigvee \Delta$

**Case.** $\mu$L

$$\frac{\mathcal{E}_1 :: \Sigma; \Gamma, \psi\{\vec{t}/\vec{x}\} \vdash \Delta \qquad \mathcal{E}_2 :: \Sigma, \vec{y}; \Gamma, \varphi\{\lambda\vec{x}.\psi/X\}\{\vec{y}/\vec{x}\} \vdash \psi\{\vec{y}/\vec{x}\}}{\Sigma; \Gamma, (\mu X(\vec{x}).\varphi)(\vec{t}) \vdash \Delta}\ \mu\text{L}$$

By assumptions, given any $\mathcal{I}$, $\theta$ where $\mathrm{dom}(\theta) = \Sigma$

$\theta;\mathcal{I} \vDash \bigwedge \Gamma \wedge \mu X(\vec{x}).\varphi\ \vec{t}$

$\theta;\mathcal{I} \vDash \mu X(\vec{x}).\varphi\ \vec{t}$

$\vec{t} \in \mu S.F^{X,\vec{x}}_{\mathcal{I},\theta}(\varphi)(S) = \mu S.\{\vec{d} \mid \theta, \vec{x} \mapsto \vec{d}; \mathcal{I}, X \mapsto S \vDash \varphi\}$ (1)

By Knaster-Tarski theorem

the least fixed point of $F$ is the intersection of pre-fixed points of $F$

$\mu S.F^{X,\vec{x}}_{\mathcal{I},\theta}(\varphi)(S) = \cap\{S \mid F^{X,\vec{x}}_{\mathcal{I},\theta}(\varphi)(S) \subseteq S\}$

By set theory

$\mu S.F^{X,\vec{x}}_{\mathcal{I},\theta}(\varphi)(S)$ is a subset of any $S$ such that $F^{X,\vec{x}}_{\mathcal{I},\theta}(\varphi)(S) \subseteq S$ (2)

Let $T = \{\vec{d} \mid \vec{x} \mapsto \vec{d}; \mathcal{I} \vDash \psi\}$,

By Lemma A.2,

$F^{X,\vec{x}}_{\mathcal{I},\theta}(\varphi)(T) = \{\vec{d} \mid \theta, \vec{x} \mapsto \vec{d}; \mathcal{I}, X \mapsto T \vDash \varphi\} = \{\vec{d} \mid \theta, \vec{x} \mapsto \vec{d}; \mathcal{I} \vDash \varphi\{\lambda\vec{x}.\psi/X\}\}$ (3)

By I.H. on $\mathcal{E}_2$,

$\forall\mathcal{I}'$, $\forall\theta'$ where $\mathrm{dom}(\theta') = \Sigma, \vec{y}$

$\theta';\mathcal{I}' \vDash \bigwedge \Gamma \wedge \varphi\{\lambda\vec{x}.\psi/X\}\{\vec{y}/\vec{x}\}$ implies $\theta';\mathcal{I}' \vDash \psi\{\vec{y}/\vec{x}\}$

$\{\vec{d} \mid \theta, \vec{y} \mapsto \vec{d}; \mathcal{I} \vDash \varphi\{\lambda\vec{x}.\psi/X\}\{\vec{y}/\vec{x}\}\} \subseteq \{\vec{d} \mid \theta, \vec{y} \mapsto \vec{d}; \mathcal{I} \vDash \psi\{\vec{y}/\vec{x}\}\}$ (4)

By (3) and (4)

$F^{X,\vec{x}}_{\mathcal{I},\theta}(\varphi)(T) \subseteq T$

By (2)

$\mu S.F^{X,\vec{x}}_{\mathcal{I},\theta}(\varphi)(S) \subseteq T = \{\vec{d} \mid \theta, \vec{x} \mapsto \vec{d}; \mathcal{I} \vDash \psi\}$ (5)

By (1) and (5)

$\theta;\mathcal{I} \vDash \psi\ \vec{t}$

By I.H. on $\mathcal{E}_1$

$\theta;\mathcal{I} \vDash \bigvee \Delta$

**Case.** $\nu$R

$$\frac{\mathcal{E}_1 :: \Sigma; \Gamma \vdash \psi\{\vec{t}/\vec{x}\}, \Delta \qquad \mathcal{E}_2 :: \Sigma, \vec{y}; \Gamma, \psi\{\vec{y}/\vec{x}\} \vdash \varphi\{\lambda\vec{x}.\psi/X\}\{\vec{y}/\vec{x}\}}{\Sigma; \Gamma \vdash (\nu X(\vec{x}).\varphi)(\vec{t}), \Delta}\ \nu\text{R}$$

By assumptions, given any $\mathcal{I}$, $\theta$ where $\mathrm{dom}(\theta) = \Sigma$

$\theta;\mathcal{I} \vDash \bigwedge \Gamma$

By I.H. on $\mathcal{E}_1$

$\theta;\mathcal{I} \vDash \psi\{\vec{t}/\vec{x}\} \vee \bigvee \Delta$

If $\theta;\mathcal{I} \vDash \Delta$ then $\theta;\mathcal{I} \vDash (\nu X(\vec{x}).\varphi)(\vec{t}) \vee \bigvee \Delta$

otherwise $\theta;\mathcal{I} \vDash \psi\{\vec{t}/\vec{x}\}$, therefore $\vec{t} \in \{\vec{d} \mid \theta, \vec{x} \mapsto \vec{d}; \mathcal{I} \vDash \psi\}$ (1)

By Knaster-Tarski theorem,

The greatest fixed point of $F$ is the union of post-fixed points of $F$

$\nu S.F^{X,\vec{x}}_{\mathcal{I},\theta}(\varphi)(S) = \cup\{S \mid S \subseteq F^{X,\vec{x}}_{\mathcal{I},\theta}(\varphi)(S)\}$

By set theory

$\nu S.F^{X,\vec{x}}_{\mathcal{I},\theta}(\varphi)(S)$ is a superset of any $S$ such that $S \subseteq F^{X,\vec{x}}_{\mathcal{I},\theta}(\varphi)(S)$ (2)

Let $T = \{\vec{d} \mid \vec{x} \mapsto \vec{d}; \mathcal{I} \vDash \psi\}$, By Lemma A.2,

$$F^{X;\vec{x}}_{\mathcal{I},\theta}(\varphi)(T) = \{\vec{d} \mid \theta, \vec{x} \mapsto \vec{d}; \mathcal{I}, X \mapsto T \vDash \varphi\} = \{\vec{d} \mid \theta, \vec{x} \mapsto \vec{d}; \mathcal{I} \vDash \varphi\{\lambda\vec{x}.\psi/X\}\} \tag{3}$$

By I.H. on $\mathcal{E}_2$,

$\forall\mathcal{I}'$, $\forall\theta'$ where $\mathrm{dom}(\theta') = \Sigma, \vec{y}$

$\theta'; \mathcal{I}' \vDash \psi\{\vec{y}/\vec{x}\}$ implies $\theta'; \mathcal{I}' \vDash \bigwedge\Gamma \wedge \varphi\{\lambda\vec{x}.\psi/X\}\{\vec{y}/\vec{x}\}$

$$\{\vec{d} \mid \theta, \vec{y} \mapsto \vec{d}; \mathcal{I} \vDash \psi\{\vec{y}/\vec{x}\}\} \subseteq \{\vec{d} \mid \theta, \vec{y} \mapsto \vec{d}; \mathcal{I} \vDash \varphi\{\lambda\vec{x}.\psi/X\}\{\vec{y}/\vec{x}\}\} \tag{4}$$

By (3) and (4)

$T \subseteq F^{X;\vec{x}}_{\mathcal{I},\theta}(\varphi)(T)$

By (2)

$$T = \{\vec{d} \mid \theta, \vec{x} \mapsto \vec{d}; \mathcal{I} \vDash \psi\} \subseteq \nu S.F^{X;\vec{x}}_{\mathcal{I},\theta}(\varphi)(S) \tag{5}$$

By (1) and (5)

$\theta; \mathcal{I} \vDash \nu X(\vec{x}).\varphi\ \vec{t}$

$\theta; \mathcal{I} \vDash \nu X(\vec{x}).\varphi\ \vec{t} \vee \bigvee\Delta$

$\square$

# C Model-Checking for LFP

This appendix proves that the model-checking tableau for LFP (Section 5.2) are sound. In order to do that, we define the interpretation $\iota_{\mathcal{I}}(\mathcal{E}, \Delta)$ generated from a list of equations $\mathcal{E}$ and a list of assumptions $\Delta$, given an interpretation of predicates $\mathcal{I}$. The interpretation is defined by induction on $\mathcal{E}$.

$$\begin{array}{lcl} \iota_{\mathcal{I}}([\,], \Delta) & = & \bullet \\ \iota_{\mathcal{I}}(\mathcal{E} :: (X \mapsto \lambda\vec{x}.\varphi), \Delta \cup \{X : S\}) & = & \mathcal{I}',\ X \mapsto \nu(\lambda S'.\ [\![\, S \,]\!]^{\bullet} \cup F^{X,\vec{x}}_{(\mathcal{I},\mathcal{I}'),\bullet}(\varphi)(S')) \quad (\text{where } \mathcal{I}' = \iota_{\mathcal{I}}(\mathcal{E}, \Delta)) \end{array}$$

Observe that since $F^{X,\vec{x}}_{(\mathcal{I},\mathcal{I}'),\bullet}(\varphi)(S')$ is a monotonic function of $S'$, it is also the case that $\lambda S'.\ [\![\, S \,]\!]^{\bullet} \cup F^{X,\vec{x}}_{(\mathcal{I},\mathcal{I}'),\bullet}(\varphi)(S')$ is a monotonic function. Hence, its greatest fixed-point in the second clause above exists.

**Lemma C.1.** *Suppose that all free predicate variables of $\varphi$ are defined in $\mathcal{E}$. Then $\theta; \mathcal{I}, \iota_{\mathcal{I}}(\mathcal{E}, \Delta) \models \varphi$ iff $\theta; \mathcal{I}, \iota_{\mathcal{I}}((\mathcal{E}, \mathcal{E}'), \Delta) \models \varphi$*

*Proof.* By induction on $\mathcal{E}'$. $\square$

**Lemma C.2** (Reduction; reproduced from [34])**.** *Let $f : 2^A \to 2^A$ be a monotonic function. Then $S \subseteq \nu f$ if and only if $S \subseteq f(\nu(\lambda S'.\ S \cup f(S')))$.*

**Lemma C.3** (Reduction for singletons)**.** *Let $f : 2^A \to 2^A$ be monotonic and suppose $d \notin S \subsetneq A$. Then, $d \in \nu(\lambda S'.\ S \cup f(S'))$ if and only if $d \in f(\nu(\lambda S'.\ S \cup \{d\} \cup f(S')))$.*

*Proof.* We have:

$$\begin{array}{lll} & d \in \nu(\lambda S'.\ S \cup f(S')) & \\ \leftrightarrow & \{d\} \subseteq \nu(\lambda S'.\ S \cup f(S')) & \\ \leftrightarrow & \{d\} \subseteq S \cup f(\nu(\lambda S'.\ S \cup \{d\} \cup f(S'))) & (\text{Lemma C.2}) \\ \leftrightarrow & d \in S \cup f(\nu(\lambda S'.\ S \cup \{d\} \cup f(S'))) & \\ \leftrightarrow & d \in f(\nu(\lambda S'.\ S \cup \{d\} \cup f(S'))) & (\text{Assumption that } d \notin S) \end{array}$$

$\square$

**Theorem C.4** (Soundness)**.** *$\sigma; \mathcal{E}; \Delta \vdash \varphi$ implies $\bullet; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta) \models \varphi$.*

*Proof.* We induct on the derivation of $\mathcal{I}; \mathcal{E}; \Delta \vdash \varphi$, and case analyze its last rule.

**Case.** $\dfrac{[\![\vec{t}]\!]^\bullet \in \mathcal{I}_\sigma(P)}{\sigma; \mathcal{E}; \Delta \vdash P\ \vec{t}}$ INIT

We need to show that $\bullet; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta) \models P\ \vec{t}$. By definition of the semantics, it suffices to show that $[\![\vec{t}]\!]^\bullet \in \mathcal{I}_\sigma(P)$, which is stated in the rule's premise.

**Case.** $\dfrac{[\![\vec{t}]\!]^\bullet \notin \mathcal{I}_\sigma(P)}{\sigma; \mathcal{E}; \Delta \vdash \neg(P\ \vec{t})}\ \neg$

We need to show that $\bullet; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta) \models \neg(P\ \vec{t})$. By definition of the semantics, it suffices to show that $[\![\vec{t}]\!]^\bullet \notin \mathcal{I}_\sigma(P)$, which is stated in the rule's premise.

**Case.** $\dfrac{}{\sigma; \mathcal{E}; \Delta \vdash \top}\ \top$

We have to show that $\bullet; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta) \models \top$, which always holds by definition of the semantics.

**Case.** $\dfrac{\sigma; \mathcal{E}; \Delta \vdash \varphi_1 \qquad \sigma; \mathcal{E}; \Delta \vdash \varphi_2}{\sigma; \mathcal{E}; \Delta \vdash \varphi_1 \wedge \varphi_2}\ \wedge$

We need to show that $\bullet; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta) \models \varphi_1 \wedge \varphi_2$ or, equivalently, $\bullet; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta) \models \varphi_1$ and $\bullet; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta) \models \varphi_2$. The latter two follow from i.h. on the premises.

**Case.** $\dfrac{\sigma; \mathcal{E}; \Delta \vdash \varphi_1}{\sigma; \mathcal{E}; \Delta \vdash \varphi_1 \vee \varphi_2}\ \vee 1$

We need to show that $\bullet; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta) \models \varphi_1 \vee \varphi_2$. By definition of the semantics, it suffices to show that $\bullet; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta) \models \varphi_1$, which follows immediately from the i.h. applied to the premise.

**Case.** $\dfrac{\sigma; \mathcal{E}; \Delta \vdash \varphi_2}{\sigma; \mathcal{E}; \Delta \vdash \varphi_1 \vee \varphi_2}\ \vee 2$

Similar to the previous case.

**Case.** $\dfrac{\text{all } d \in \mathrm{D}.\ (\sigma; \mathcal{E}; \Delta \vdash \varphi\{d/x\})}{\sigma; \mathcal{E}; \Delta \vdash \forall x.\varphi}\ \forall$

We need to show that $\bullet; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta) \models \forall x.\varphi$. By definition of the semantics, it suffices to show that for any $d \in \mathrm{D}$, $\bullet; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta) \models \varphi\{d/x\}$. So pick any $d \in \mathrm{D}$. By i.h. on the premise, $\bullet; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta) \models \varphi\{d/x\}$, as required.

**Case.** $\dfrac{d \in \mathrm{D} \qquad \sigma; \mathcal{E}; \Delta \vdash \varphi\{d/x\}}{\sigma; \mathcal{E}; \Delta \vdash \exists x.\varphi}\ \exists$

We need to show that $\bullet; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta) \models \exists x.\varphi$. By i.h. on the premise, $\bullet; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta) \models \varphi\{d/x\}$. By definition of semantics, $\bullet; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta) \models \exists x.\varphi$, as required.

**Case.** $\dfrac{\sigma; \mathcal{E}; \Delta \vdash \varphi\{\vec{t}/\vec{x}\}\{(\mu X(\vec{x}).\ \varphi)/X\}}{\sigma; \mathcal{E}; \Delta \vdash (\mu X(\vec{x}).\ \varphi)\ \vec{t}}\ \mu$

We need to show that $\bullet; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta) \models (\mu X(\vec{x}).\ \varphi)\ \vec{t}$. By i.h. on the premise we have $\bullet; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta) \models \varphi\{\vec{t}/\vec{x}\}\{(\mu X(\vec{x}).\ \varphi)/X\}$. Now we have:

$$\begin{array}{lll} & \bullet; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta) \models \varphi\{\vec{t}/\vec{x}\}\{(\mu X(\vec{x}).\ \varphi)/X\} & \\ \leftrightarrow & \vec{x} \mapsto [\![\vec{t}\,]\!]^\bullet; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta) \models \varphi\{(\mu X(\vec{x}).\ \varphi)/X\} & \text{(Lemma A.1)} \\ \leftrightarrow & \bullet; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta) \models (\mu X(\vec{x}).\ \varphi)\ \vec{t} & \text{(Lemma A.3)} \end{array}$$

**Case.** $\dfrac{\sigma; \mathcal{E}, X \Rightarrow \lambda\vec{x}.\varphi; \Delta, X : \{\} \vdash X\ \vec{t} \qquad (X \text{ fresh})}{\sigma; \mathcal{E}; \Delta \vdash (\nu X(\vec{x}).\ \varphi)\ \vec{t}}\ \nu$

We need to show that $\bullet; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta) \models (\nu X(\vec{x}).\ \varphi)\ \vec{t}$. Let $\mathcal{E}' = \mathcal{E}, X \Rightarrow \lambda\vec{x}.\varphi$ and $\Delta' = \Delta, X : \{\}$. Then,

$$\begin{array}{lll} & \bullet; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}', \Delta') \models X\ \vec{t} & \text{(i.h.)} \\ \leftrightarrow & [\![\vec{t}\,]\!]^\bullet \in \iota_{\mathcal{I}_\sigma}(\mathcal{E}', \Delta')(X) & \text{(Defn. of } \theta; \mathcal{I}_\sigma \models \cdot) \\ \leftrightarrow & [\![\vec{t}\,]\!]^\bullet \in \nu(\lambda S'.\ \{\} \cup F^{X,\vec{x}}_{(\mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta')), \bullet}(\varphi)(S')) & \text{(Defn. of } \iota_{\mathcal{I}_\sigma}(\mathcal{E}', \Delta') \text{ and } X : \{\} \in \Delta') \\ \leftrightarrow & [\![\vec{t}\,]\!]^\bullet \in \nu(\lambda S'.\ F^{X,\vec{x}}_{(\mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta')), \bullet}(\varphi)(S')) & \\ \leftrightarrow & [\![\vec{t}\,]\!]^\bullet \in \nu(F^{X,\vec{x}}_{(\mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta')), \bullet}(\varphi)) & (\lambda x.(f\ x) = f) \\ \leftrightarrow & [\![\vec{t}\,]\!]^\bullet \in \nu(F^{X,\vec{x}}_{(\mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta)), \bullet}(\varphi)) & (\text{Defined-variables}(\mathcal{E}) \subseteq \Delta) \\ \leftrightarrow & \bullet; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta) \models (\nu X(\vec{x}).\ \varphi)\ \vec{t} & \text{(Defn. of } \theta; \mathcal{I}_\sigma \models (\nu X(\vec{x}).\ \varphi)\ \vec{t}) \end{array}$$

**Case.** $\dfrac{[\![\vec{t}\,]\!]^\bullet \in [\![S]\!]^\bullet}{\sigma; \mathcal{E}; \Delta, X : S \vdash X\ \vec{t}}\ X1$

We want to show that $\bullet; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, (\Delta, X : S)) \models X\ \vec{t}$ or, equivalently, that $[\![\vec{t}\,]\!]^\bullet \in \iota_{\mathcal{I}_\sigma}(\mathcal{E}, (\Delta, X : S))(X)$. Since some definition for $X$ must exist in $\mathcal{E}$, suppose that $\mathcal{E} = \mathcal{E}_1, (X \Rightarrow \lambda\vec{x}.\ \varphi), \mathcal{E}_2$. Then, by definition of $\iota_{\mathcal{I}_\sigma}$ we have: $\iota_{\mathcal{I}_\sigma}(\mathcal{E}, (\Delta, X : S))(X) = \nu(\lambda S'.\ [\![S]\!]^\bullet \cup F^{X,\vec{x}}_{(\mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}_1, (\Delta, X:S))), \bullet}(\varphi)(S'))$. Hence, it suffices to show that $[\![\vec{t}\,]\!]^\bullet \in \nu(\lambda S'.\ [\![S]\!]^\bullet \cup F^{X,\vec{x}}_{(\mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}_1, (\Delta, X:S))), \bullet}(\varphi)(S'))$. To avoid syntactic clutter, define $f = F^{X,\vec{x}}_{(\mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}_1, (\Delta, X:S))), \bullet}(\varphi)$. What we need to show then is that: $[\![\vec{t}\,]\!]^\bullet \in \nu(\lambda S'.\ [\![S]\!]^\bullet \cup f(S'))$. Suppose $\nu(\lambda S'.\ [\![S]\!]^\bullet \cup f(S')) = S_0$. Because $S_0$ is a fixed-point of $\lambda S'.\ [\![S]\!]^\bullet \cup f(S')$, it follows that $S_0 = [\![S]\!]^\bullet \cup f(S_0) \supseteq [\![S]\!]^\bullet$. Since, by the premise of the rule, $[\![\vec{t}\,]\!]^\bullet \in [\![S]\!]^\bullet$, it follows that $[\![\vec{t}\,]\!]^\bullet \in S_0 = \nu(\lambda S'.\ [\![S]\!]^\bullet \cup f(S'))$, as required.

**Case.** $\dfrac{[\![\vec{t}\,]\!]^\bullet \notin [\![S]\!]^\bullet \qquad (X \Rightarrow \lambda\vec{x}.\varphi) \in \mathcal{E} \qquad \sigma; \mathcal{E}; \Delta, X : S \cup \{\vec{t}\} \vdash \varphi\{\vec{t}/\vec{x}\}}{\sigma; \mathcal{E}; \Delta, X : S \vdash X\ \vec{t}}\ X2$

Let $\Delta' = \Delta, X : S$ and let $\mathcal{E} = \mathcal{E}_1, (X \Rightarrow \lambda\vec{x}.\varphi), \mathcal{E}_2$. We want to show that $\bullet; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta') \models X\ \vec{t}$. We have:

$$\begin{array}{lll} & \bullet; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta') \models X\ \vec{t} & \\ \leftrightarrow & [\![\vec{t}\,]\!]^\bullet \in \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta')(X) & \text{(Defn. of } \theta; \mathcal{I}_\sigma \models \cdot) \\ \leftrightarrow & [\![\vec{t}\,]\!]^\bullet \in \nu(\lambda S'.\ [\![S]\!]^\bullet \cup F^{X,\vec{x}}_{(\mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}_1, \Delta')), \bullet}(\varphi)(S')) & \text{(Defn. of } \iota_{\mathcal{I}_\sigma}(\mathcal{E}, \Delta')) \\ \leftrightarrow & [\![\vec{t}\,]\!]^\bullet \in \nu(\lambda S'.\ [\![S]\!]^\bullet \cup F^{X,\vec{x}}_{(\mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}_1, \Delta)), \bullet}(\varphi)(S')) & (\text{Defined-variables}(\mathcal{E}_1) \subseteq \Delta) \\ \leftrightarrow & [\![\vec{t}\,]\!]^\bullet \in F^{X,\vec{x}}_{(\mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}_1, \Delta)), \bullet}(\varphi)(\nu(\lambda S'.\ [\![S]\!]^\bullet \cup \{[\![\vec{t}\,]\!]^\bullet\} \cup F^{X,\vec{x}}_{(\mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}_1, \Delta)), \bullet}(\varphi)(S'))) & \text{(Lemma C.3)} \\ \leftrightarrow & \vec{x} \mapsto \vec{t}; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}_1, \Delta), X \mapsto \nu(\lambda S'.\ [\![S]\!]^\bullet \cup \{[\![\vec{t}\,]\!]^\bullet\} \cup F^{X,\vec{x}}_{(\mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}_1, \Delta)), \bullet}(\varphi)(S')) \models \varphi & \text{(Defn. of } F^{X,\vec{x}}_{\mathcal{I}_\sigma, \theta}(\varphi)) \\ \leftrightarrow & \vec{x} \mapsto [\![\vec{t}\,]\!]^\bullet; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}((\mathcal{E}_1, (X \Rightarrow \lambda\vec{x}.\varphi)), (\Delta, X : S \cup \{\vec{t}\})) \models \varphi & \text{(Defn. of } \iota) \\ \leftrightarrow & \vec{x} \mapsto [\![\vec{t}\,]\!]^\bullet; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, (\Delta, X : S \cup \{\vec{t}\})) \models \varphi & \text{(Lemma C.1)} \\ \leftrightarrow & \bullet; \mathcal{I}_\sigma, \iota_{\mathcal{I}_\sigma}(\mathcal{E}, (\Delta, X : S \cup \{\vec{t}\})) \models \varphi\{\vec{t}/\vec{x}\} & \text{(Lemma A.1)} \end{array}$$

The last statement follows from the i.h. So the first statement must also hold.

□

**Theorem C.5** (Soundness)**.** *Suppose $\varphi$ has no free term and predicate variables and $\sigma; \bullet; \bullet \vdash \varphi$. Then, $\bullet; \sigma \models \varphi$.*

*Proof.* Suppose $\sigma; \bullet; \bullet \vdash \varphi$. By Theorem C.4, $\bullet; \mathcal{I}_\sigma \models \varphi$. By definition of satisfaction on traces, this is the same as $\bullet; \sigma \models \varphi$. □

# D Local Feasibility

## D.1 Formal Definitions

**Strict Past Operator** We define a strict past operator $\diamondminus_{st}\ \phi$ as follows. Formula $\diamondminus_{st}\ \phi$ holds in the current state if $\phi$ holds in some state that is strictly earlier than the current state.

$$(\diamondminus_{st}\ \phi)^{@s} \quad \triangleq \quad \exists s'. (s' <_{\mathsf{st}} s) \wedge \phi^{@s'}$$

**Trace Operations** Given $f$ and $g$, both of which are mappings from domain $D_1$ to $D_2$, we define $f \cup g$ as follows:

$$(f \cup g)(d) = \begin{cases} f(d) \cup g(d) & \text{if } d \in \text{dom}(f) \cap \text{dom}(g) \\ f(d) & \text{if } d \in \text{dom}(f) \text{ and } d \notin \text{dom}(g) \\ g(d) & \text{if } d \notin \text{dom}(f) \text{ and } d \in \text{dom}(g) \end{cases}$$

Given $f$ and $g$, both of which are mappings from domain $D_1$ to $D_2$, we define $f \sqcup g$ as follows:

$$(f \sqcup g)(d) = \begin{cases} f(d) \cup g(d) & \text{if } d \in \text{dom}(f) \cap \text{dom}(g) \text{ and } f(d) = g(d) \\ f(d) & \text{if } d \in \text{dom}(f) \text{ and } d \notin \text{dom}(g) \\ g(d) & \text{if } d \notin \text{dom}(f) \text{ and } d \in \text{dom}(g) \end{cases}$$

Given $f$ and $g$, both of which are mappings from domain $D_1$ to mappings from $D_2$ to $D_3$, we define $f \sqcup\!\!\!\sqcup g$ as follows:

$$(f \sqcup\!\!\!\sqcup g)(d) = \begin{cases} f(d) \sqcup g(d) & \text{if } d \in \text{dom}(f) \cap \text{dom}(g) \\ f(d) & \text{if } d \in \text{dom}(f) \text{ and } d \notin \text{dom}(g) \\ g(d) & \text{if } d \notin \text{dom}(f) \text{ and } d \in \text{dom}(g) \end{cases}$$

Given $f$ and $g$, both of which are mappings from domain $D_1$ to mappings from $D_2$ to $D_3$, we define $f \Cup g$ as follows:

$$(f \Cup g)(d) = \begin{cases} f(d) \cup g(d) & \text{if } d \in \text{dom}(f) \cap \text{dom}(g) \\ f(d) & \text{if } d \in \text{dom}(f) \text{ and } d \notin \text{dom}(g) \\ g(d) & \text{if } d \notin \text{dom}(f) \text{ and } d \in \text{dom}(g) \end{cases}$$

$\widehat{\sigma}_1 = (\kappa_1, \rho_1^A, \rho_1^B, a_1, \neg a_1, \tau_1, \iota_1)$, $\widehat{\sigma}_2 = (\kappa_2, \rho_1^A, \rho_2^B, a_2, \neg a_2, \tau_2, \iota_2)$.
$\widehat{\sigma}_1 \uplus \widehat{\sigma}_2 = (\kappa_1 \sqcup\!\!\!\sqcup \kappa_2, \rho_1^A \sqcup\!\!\!\sqcup \rho_2^A, \rho_1^B \Cup \rho_2^B, a_1 \cup a_2, \neg a_1 \cup \neg a_2, \tau_1 \sqcup \tau_2, \iota_1 \cup \iota_2)$

We say a mapping $f$ from domain $D1$ to $D2$ is an extension of $g$ ($f \supseteq g$), if $\forall d \in \text{dom}(g)$, $g(d) \subseteq f(d)$. We say a mapping $ff$ from domain $D1$ to a mapping from $D2$ to $D3$ is an extension of $gg$, if $\forall d \in dom(gg)$, $gg(d) \subseteq\ ff(d)$.

We say a trace $\widehat{\sigma}$ is an extension of $\widehat{\sigma}'$, written $\widehat{\sigma} \supseteq \widehat{\sigma}'$, if any mapping $f$ in $\widehat{\sigma}$ is the extension of the corresponding mapping $f'$ in $\widehat{\sigma}'$.

We write $\sigma|_p^{\mathcal{A}} = \emptyset$ to mean that $p$ is the performer of any of the actions in the range of function $a$.

We assume that each agent has some default knowledge. We write $\kappa^p$ to denote a knowledge map where $\kappa^p(i)(p)$ is $p$'s default knowledge, for each $i$ in the domain of $\kappa^p$.

**Feasibility** Our definitions for feasibility rely on a recursively defined function ($GF(j, \Phi, Sa)$) to generate conditions that need to be satisfied at each state in an infinite run of the system (Figure 6). Each $GF(j, \Phi, Sa)$ contains a hole ([ ]).We write $GF(j, \Phi, Sa)[R]_{\vec{x}}$ to denote the predicate generated by unfolding the definition of $GF$ and plug $R$ in the hole. The variables annotated in the subscripts of the hold appear free in $R$, and are bound by quantifiers on the outer layers.

Ultimately, we care about $GF(j, \Phi, Sa)[\mathsf{true}]$.

$$
\begin{array}{lcl}
H(\widehat{\sigma}, i, Sa, t_0) & = & (\widehat{\sigma}\,|_{i-1} = \emptyset) \wedge (\widehat{\sigma}\,|_{i} = \widehat{\sigma}) \wedge \mathit{start}(\tau) = t_0 \wedge \forall p \in Sa, (\widehat{\sigma}\,|_p^{\mathcal{A}} = \emptyset) \wedge \forall p \in Sa, \kappa \supseteq \kappa^p \\
F(\widehat{\sigma}, i, Sa, t_0) & = & \forall P \in \text{range}(a), \exists p \in Sa \text{ such that } p \text{ is the performer of } P \\
 & & \forall P \in \text{range}(\neg a), \exists p \in Sa \text{ such that } p \text{ is the performer of } P \\
 & & \forall k \in \text{dom}(\widehat{\sigma}), k \geq i, \mathit{start}(\tau) = t_0
\end{array}
$$

$$
\begin{array}{lcl}
GF(0, \Phi, Sa) & = & \forall \widehat{\sigma}_0, H(\widehat{\sigma}_0, 0, Sa, \tau_0(0)) \supset \\
 & & \quad \exists \widehat{\sigma}_0^{Sa}, F(\widehat{\sigma}_0^{Sa}, 0, Sa, \tau_0(0) \wedge \widehat{\sigma}_0 \uplus \widehat{\sigma}_0^{Sa} \text{ is well-defined} \\
 & & \quad\quad \forall \widehat{\sigma}', \widehat{\sigma}'\,|_0 = \emptyset \wedge \widehat{\sigma}' \uplus \widehat{\sigma}_0 \uplus \widehat{\sigma}_0^{Sa} \text{ is well-defined} \supset \\
 & & \quad\quad\quad \forall\, \mathbf{G}\, \varphi_i \in \Phi, \mathit{Tr}(\widehat{\sigma}' \uplus \widehat{\sigma}_0 \uplus \widehat{\sigma}_0^{Sa}), 0 \vDash \varphi_i \\
 & & \quad\quad\quad \wedge [\ ]_{\widehat{\sigma}_0, \widehat{\sigma}_0^{Sa}}
\end{array}
$$

$$
\begin{array}{l}
GF(j, \Phi, Sa) \quad = \\
\quad GF(j-1, \Phi, Sa)[\forall \widehat{\sigma}_j, H(\widehat{\sigma}_j, j, Sa, \tau_0(0)) \supset \\
\qquad\qquad (\exists \widehat{\sigma}_j^{Sa}, F(\widehat{\sigma}_j^{Sa}, j, Sa, \tau_0(0) \wedge \biguplus_{k=0}^{j} \widehat{\sigma}_k \uplus \widehat{\sigma}_k^{Sa} \text{ is well-defined} \\
\qquad\qquad \forall \widehat{\sigma}', \widehat{\sigma}'\,|_j = \emptyset \wedge \widehat{\sigma}' \uplus \biguplus_{k=0}^{j} \widehat{\sigma}_k \uplus \widehat{\sigma}_k^{Sa} \text{ is well-defined} \supset \\
\qquad\qquad\quad \forall\, \mathbf{G}\, \varphi_i \in \Phi, \mathit{Tr}(\widehat{\sigma}' \uplus \biguplus_{k=0}^{j} \widehat{\sigma}_k \uplus \widehat{\sigma}_k^{Sa}), j \vDash \varphi_i \\
\qquad\qquad \wedge [\ ]_{\widehat{\sigma}_0, \widehat{\sigma}_0^{Sa}, \cdots \widehat{\sigma}_{j-1}, \widehat{\sigma}_{j-1}^{Sa}} \\
\qquad\quad ]_{\widehat{\sigma}_0, a_0, \neg a_0, \tau_0, \cdots \widehat{\sigma}_{j-1}, \widehat{\sigma}_0^{Sa}, \cdots \widehat{\sigma}_j, \widehat{\sigma}_j^{Sa}}
\end{array}
$$

Figure 6: Feasibility

**Def.** A set of responsibilities $\Phi$ is feasible for a set of agents $Sa$ if for all $j$ iff $GF(j, \Phi, Sa)[\mathsf{true}]$.

## D.2 Lemmas and Definitions for Proving Local Feasibility Theorems

### D.2.1 Locality

**Def. Observable** We say that $\varphi$ is observable to an agent $p$ is all the atomic formulas in $\varphi$ describe states or events observable to $p$. We write $p \vdash \varphi\ \mathsf{Obs}$ to mean that $\varphi$ describes states and events that are observable by $p$. Figure 7 shows the summary of rules.

**Theorem D.1.** *If $p \vdash \varphi\ \mathsf{Obs}$, then $\varphi$ is local to agent $p$.*

**Proof** (sketch): By induction on the structure of the derivation $p \vdash \varphi\ \mathsf{Obs}$. □

### D.2.2 Lemmas and Definitions Related to Past Formulas

We define $\varphi_{past}$ to be temporal formulas that do not contain future operators. We prove the following lemma, which states that given any state $i$, $\varphi_{past}$ does not concern any states that is later than $i$.

**Lemma D.2** (Invariant of Past Formula (prefix))**.** *For all $j \geq i$, $\sigma, i \vDash \varphi_{past}$ iff $\sigma\,|_j, i \vDash \varphi_{past}$*

**Proof** (sketch): By induction on the structure of $\varphi_{past}$. We show a few key cases below.

$\boxed{p \vdash \varphi \ \mathsf{Obs}}$

$$\frac{}{p \vdash \bot \ \mathsf{Obs}} \qquad \frac{}{p \vdash \top \ \mathsf{Obs}} \qquad \frac{P \text{ is observable to } p}{p \vdash P \ \mathsf{Obs}} \qquad \frac{p \vdash \varphi_1 \ \mathsf{Obs} \qquad p \vdash \varphi_2 \ \mathsf{Obs}}{p \vdash \varphi_1 \wedge \varphi_2 \ \mathsf{Obs}}$$

$$\frac{p \vdash \varphi_1 \ \mathsf{Obs} \qquad p \vdash \varphi_2 \ \mathsf{Obs}}{p \vdash \varphi_1 \vee \varphi_2 \ \mathsf{Obs}} \qquad \frac{p \vdash \varphi \ \mathsf{Obs}}{p \vdash \exists x.\varphi \ \mathsf{Obs}} \qquad \frac{p \vdash \varphi \ \mathsf{Obs}}{p \vdash \forall x.\varphi \ \mathsf{Obs}} \qquad \frac{p \vdash \varphi \ \mathsf{Obs}}{p \vdash \neg\varphi \ \mathsf{Obs}} \qquad \frac{p \vdash \varphi \ \mathsf{Obs}}{p \vdash \diamondminus\varphi \ \mathsf{Obs}}$$

$$\frac{p \vdash \varphi \ \mathsf{Obs}}{p \vdash \diamondminus_{st}\, \varphi \ \mathsf{Obs}} \qquad \frac{p \vdash \varphi \ \mathsf{Obs}}{p \vdash \boxminus\varphi \ \mathsf{Obs}} \qquad \frac{p \vdash \varphi_1 \ \mathsf{Obs} \qquad p \vdash \varphi_2 \ \mathsf{Obs}}{p \vdash \varphi_1 \,\mathcal{S}\, \varphi_2 \ \mathsf{Obs}} \qquad \frac{p \vdash \varphi \ \mathsf{Obs}}{p \vdash \Diamond\varphi \ \mathsf{Obs}} \qquad \frac{p \vdash \varphi \ \mathsf{Obs}}{p \vdash \Box\varphi \ \mathsf{Obs}}$$

$$\frac{p \vdash \varphi_1 \ \mathsf{Obs} \qquad p \vdash \varphi_2 \ \mathsf{Obs}}{p \vdash \varphi_1 \,\mathcal{U}\, \varphi_2 \ \mathsf{Obs}}$$

Figure 7: Observable Formulas

Case $\varphi_{past} = P(\vec{t})$
The validity of the atomic predicate $P(\vec{t})$,
depending on the following functions $a(i)$, $\kappa(i)$, $\rho(i)$ and $\iota(i)$,
When $j \geq i$, the projection does not affect those mappings.
Therefore, the conclusion holds.

Case $\varphi_{past} = \neg\varphi_{past1}$
The if direction
By assumption,
$\sigma, i \vDash \neg\varphi_{past1}$
$\sigma, i \nvDash \varphi_{past1}$
By I.H. on $\varphi_{past1}$,
$\sigma|_j, i \nvDash \varphi_{past1}$
$\sigma|_j, i \vDash \neg\varphi_{past1}$
The only if direction
By assumption,
$\sigma|_j, i \vDash \neg\varphi_{past1}$
$\sigma|_j, i \nvDash \varphi_{past1}$
By I.H. on $\varphi_{past1}$,
$\sigma, i \nvDash \varphi_{past1}$
$\sigma, i \vDash \neg\varphi_{past1}$

Case $\varphi_{past} = \boxminus\varphi_{past1}$
The if direction
By assumption,
$\sigma, i \vDash \boxminus\varphi_{past1}$
By the definition of $\vDash$, for all $k \leq i$,
$\sigma, k \vDash \varphi_{past1}$
$k \leq i \leq j$
By I.H. on $\varphi_{past1}$,
$\sigma|_j, k \vDash \varphi_{past1}$
By the definition of $\vDash$,
$\sigma|_j, i \vDash \boxminus\varphi_{past1}$
The only if direction
By assumption,
$\sigma|_j, i \vDash \boxminus\varphi_{past1}$
By the definition of $\vDash$, for all $k \leq i$,
$\sigma|_j, k \vDash \varphi_{past1}$
$k \leq i \leq j$
By I.H. on $\varphi_{past1}$,
$\sigma, k \vDash \varphi_{past1}$
By the definition of $\vDash$,
$\sigma, i \vDash \boxminus\varphi_{past1}$

$\square$

**Def.** We write $\varphi_p$ to denote persistent past formulas, which are formulas that are true in all future state of $i$ once it becomes true in state $i$. It is defined as follows. We use two auxiliary definitions $P_p$, which denotes atomic predicates that are true in all states; and $P_{sp}$, which denotes all other predicates.

$$\begin{array}{llcl}
\text{Persistent Form} & P_p & ::= & \text{contains}(m, \cdots) \mid \cdots \\
\text{Persistent Effect Form} & P_{sp} & ::= & A^p \mid \text{inrole}(p, r) \mid \cdots \\
\text{Persistent Past Form} & \varphi_p & ::= & \top \mid P_P \mid \varphi_p \wedge \varphi_p \mid \varphi_p \vee \varphi_p \mid \forall x.\varphi_p \mid \exists x.\varphi_p \mid \diamondminus P_{sp} \mid \diamondminus_{st}\, P_{sp} \\
 & & \mid & \diamondminus\varphi_p \mid \diamondminus_{st}\, \varphi_p \mid \varphi_p \,\mathcal{S}\, \varphi_{past}
\end{array}$$

**Lemma D.3** (Persistent Past)**.**

$\sigma, i \vDash \varphi_p$ *implies* $\sigma, i \vDash \Box\varphi_p$

**Proof** (sketch): By induction on the structure of $\varphi_p$ $\square$

Judgment $\vec{x} \vdash \varphi\ \mathsf{fin}$ holds on a past formula $\varphi$ if variables $\vec{x}$ are free in formula $\varphi$, and given any state on a trace, there is only finite number of substitutions $\delta$ for $\vec{x}$ such that $\delta\varphi$ is valid in that state. We define rules for this judgment in Figure 8.

$\boxed{\vec{x} \vdash \varphi\ \mathsf{fin}}$

$$\frac{\begin{array}{c}\mathrm{fv}(\vec{t}) \supseteq \vec{x} \text{ and for all } \sigma \text{, for all } i \text{, the set of grounding substitutions } \Delta \text{ for } \mathrm{fv}(\vec{t}) \\ \text{such that for all } \delta \in \Delta,\ \sigma, i \vDash \delta(P\ \vec{x}) \text{ is finite}\end{array}}{\vec{x} \vdash P\ \vec{t}\ \mathsf{fin}}$$

$$\frac{\vec{x}_1 \vdash \varphi_1\ \mathsf{fin} \qquad \vec{x}_2 \vdash \varphi_2\ \mathsf{fin} \qquad \vec{x} = \vec{x}_1 \cup \vec{x}_2}{\vec{x} \vdash \varphi_1 \wedge \varphi_2\ \mathsf{fin}} \qquad \frac{\vec{x} \vdash \varphi_1\ \mathsf{fin} \qquad \vec{x} \vdash \varphi_2\ \mathsf{fin}}{\vec{x} \vdash \varphi_1 \vee \varphi_2\ \mathsf{fin}} \qquad \frac{\vec{x}, y \vdash \varphi\ \mathsf{fin}}{\vec{x} \vdash \exists y.\varphi\ \mathsf{fin}} \qquad \frac{\vec{x} \vdash \varphi\ \mathsf{fin}}{\vec{x} \vdash \diamondminus \varphi\ \mathsf{fin}}$$

$$\frac{\vec{x} \vdash \varphi\ \mathsf{fin}}{\vec{x} \vdash \boxminus \varphi\ \mathsf{fin}} \qquad \frac{\vec{x} \vdash \varphi_2\ \mathsf{fin} \qquad \vdash \varphi_1\ \mathsf{past}}{\vec{x} \vdash \varphi_1\ \mathcal{S}\ \varphi_2\ \mathsf{fin}} \qquad \frac{\vec{x} \vdash \varphi\ \mathsf{fin}}{\vec{x} \vdash \diamondminus_{st}\ \varphi\ \mathsf{fin}}$$

Figure 8: Judgment for Past Formulas with Finite Substitutions

**Lemma D.4** (Finite Substitution)**.** *If* $\vec{x} \vdash \varphi\ \mathsf{fin}$*, then for all* $\sigma$*, for all* $i$*, the set of all grounding substitutions* $\delta$ *for* $\vec{x}$*, such that* $\sigma, i \vDash \delta\varphi$ *is finite.*

**Proof** (sketch): By induction on the derivation $\vec{x} \vdash \varphi\ \mathsf{fin}$. $\square$

**Def. Strict Past** In order for the actions from different agents to compose nicely, we need to make sure that an agent $p$'s plan is not affected by the changes in the current state caused by another $b$. Otherwise, we would not be able to achieve a stable system. We define a syntactic check on a past formula $p \vdash \varphi\ \mathsf{StrictPast}$. The definitions are in Figure 9.

$\boxed{p \vdash \varphi\ \mathsf{StrictPast}}$

$$\frac{}{p \vdash \bot\ \mathsf{StrictPast}} \qquad \frac{}{p \vdash \top\ \mathsf{StrictPast}} \qquad \frac{P = A^p \text{ or } P \text{ depends on the default knowledge of } p}{p \vdash P\ \mathsf{StrictPast}}$$

$$\frac{p \vdash \varphi_1\ \mathsf{StrictPast} \qquad p \vdash \varphi_2\ \mathsf{StrictPast}}{p \vdash \varphi_1 \wedge \varphi_2\ \mathsf{StrictPast}} \qquad \frac{p \vdash \varphi_1\ \mathsf{StrictPast} \qquad p \vdash \varphi_2\ \mathsf{StrictPast}}{p \vdash \varphi_1 \vee \varphi_2\ \mathsf{StrictPast}} \qquad \frac{p \vdash \varphi\ \mathsf{StrictPast}}{p \vdash \exists x.\varphi\ \mathsf{StrictPast}}$$

$$\frac{p \vdash \varphi\ \mathsf{StrictPast}}{p \vdash \forall x.\varphi\ \mathsf{StrictPast}} \qquad \frac{p \vdash \varphi\ \mathsf{StrictPast}}{p \vdash \neg\varphi\ \mathsf{StrictPast}} \qquad \frac{p \vdash \varphi\ \mathsf{StrictPast}}{p \vdash \diamondminus \varphi\ \mathsf{StrictPast}} \qquad \frac{\varphi \text{ is a past formula}}{p \vdash \diamondminus_{st}\ \varphi\ \mathsf{StrictPast}}$$

$$\frac{p \vdash \varphi\ \mathsf{StrictPast}}{p \vdash \boxminus \varphi\ \mathsf{StrictPast}} \qquad \frac{p \vdash \varphi_1\ \mathsf{StrictPast} \qquad p \vdash \varphi_2\ \mathsf{StrictPast}}{p \vdash \varphi_1\ \mathcal{S}\ \varphi_2\ \mathsf{StrictPast}}$$

Figure 9: Strictly Past

**Lemma D.5** (Invariant of Strictly Past Formula).
*If $p \vdash \varphi$ $\mathsf{StrictPast}$, given any $\sigma$ and $\sigma'$, such that*

1. $\kappa \supseteq \kappa^p$ *and* $\kappa' \supseteq \kappa^p$
2. *$p$ is not the performer of any of the actions in the range of $a'$,*
3. $\sigma'|_{i-1} = \emptyset$
4. *$\sigma \uplus \sigma'$ is well-defined*

*then $\sigma, i \vDash \varphi$ iff $\sigma \uplus \sigma', i \vDash \varphi$.*

**Proof** (sketch): By induction on the structure of $a \vdash \varphi$ $\mathsf{StrictPast}$. We show a few key cases below.

Case: $\varphi = P(\vec{t})$.
If direction
By assumption,
$\sigma, i \vDash P(\vec{t})$
$P(\vec{t}) = A^p \in a(i)$, or $P(\vec{t})$ is justified by $\kappa^p$
the above conditions still hold for $\sigma \uplus \sigma'$
therefore, $\sigma \uplus \sigma', i \vDash P(\vec{t})$
Only if direction
By assumption,
$\sigma \uplus \sigma', i \vDash P(\vec{t})$
$P(\vec{t}) = A^p \in a(i) \cup a'(i)$, or $P(\vec{t})$ is justified by $\kappa^p$
By assumption, $p$ is not the performer of any actions in $a'(i)$
$P(\vec{t}) \in a(i)$, or $P(\vec{t})$ is justified by $\kappa^p$
therefore, $\sigma, i \vDash P(\vec{t})$

Case: $\dfrac{\varphi \text{ is a past formula}}{p \vdash \diamondminus_{st}\, \varphi \ \mathsf{StrictPast}}$
By assumption,
$$\sigma'|_{i-1} = \emptyset \quad (1)$$
By (1) and the definition of projection,
$$\forall k < i, \sigma|_k = (\sigma \uplus \sigma')|_k \quad (2)$$
By Lemma D.2,
$$\sigma|_k, k \vDash \varphi \text{ iff } \sigma, k \vDash \varphi \quad (3)$$
By Lemma D.2,
$$\sigma \uplus \sigma'|_k, k \vDash \varphi \text{ iff } \sigma \uplus \sigma', k \vDash \varphi \quad (4)$$
By (2),(3) and (4),
$$\sigma, k \vDash \varphi \text{ iff } \sigma \uplus \sigma', k \vDash \varphi \quad (5)$$
By (5),
$$\sigma, i \vDash \diamondminus_{st}\, \varphi \text{ iff } \sigma \uplus \sigma', i \vDash \diamondminus_{st}\, \varphi \quad (6)$$

□

**Def.** Judgment $\{P_1, \cdots, P_n\} \vdash \varphi_{past}$ holds when the validity of predicates $P_1$ to $P_n$ in a state $i$ are irrelevant to the validity of $\varphi_{past}$ in that state. It is defined in Figure 10.

**Lemma D.6** (Invariant of Past Formulas).
*If $SP \vdash \varphi$, given $\sigma$, $\sigma'$ such that*

1. $\sigma'|_{i-1} = \emptyset$,
2. *$j \geq i$, for all $P \in a'(j)$, exists $P' \in SP$ and $\delta$ such that $\delta P = \delta P'$,*
3. *$\sigma \uplus \sigma'$ is well-defined*

$\boxed{\{P_1, \cdots, P_n\} \vdash \varphi}$

$$\frac{}{SP \vdash \top} \qquad \frac{}{SP \vdash \bot} \qquad \frac{\nexists \delta \text{ st. } \delta P_i = \delta A^p}{\{P_1, \cdots, P_n\} \vdash A^p} \qquad \frac{SP \vdash \varphi}{SP \vdash \neg\varphi} \qquad \frac{SP \vdash \varphi_1 \qquad SP \vdash \varphi_2}{SP \vdash \varphi_1 \wedge \varphi_2}$$

$$\frac{SP \vdash \varphi_1 \qquad SP \vdash \varphi_2}{SP \vdash \varphi_1 \vee \varphi_2} \qquad \frac{SP \vdash \varphi}{SP \vdash \exists x.\varphi} \qquad \frac{SP \vdash \varphi}{SP \vdash \forall x.\varphi} \qquad \frac{SP \vdash \varphi}{SP \vdash \diamondminus\varphi} \qquad \frac{SP \vdash \varphi}{SP \vdash \boxminus\varphi} \qquad \frac{\varphi \text{ is a past formula}}{SP \vdash \diamondminus_{st}\, \varphi}$$

$$\frac{SP \vdash \varphi_1 \qquad SP \vdash \varphi_2}{SP \vdash \varphi_1 \,\mathcal{S}\, \varphi_2}$$

Figure 10: Irrelevant Past

*then* $\sigma, i \vDash \varphi$ *iff* $\sigma \uplus \sigma', i \vDash \varphi$.

**Proof** (sketch): By induction on the structure of the derivation $SP \vdash \varphi$.

Case: $\varphi = A^p$.

If direction

By assumption,

$\sigma, i \vDash A^p$

$A^p \in a(i)$

$A^p \in a(i) \cup a'(i)$

therefore, $\sigma \uplus \sigma', i \vDash A^p$

Only if direction

By assumption,

$\sigma \uplus \sigma', i \vDash A^p$

$A^p \in a(i) \cup a'(i)$

By assumption,

$A^p \notin a'(i)$

$A^p \in a(i)$

therefore, $\sigma, i \vDash A^p$

Case: $\dfrac{\varphi \text{ is a past formula}}{SP \vdash \diamondminus_{st}\, \varphi}$

By assumption,

$\sigma'|_{i-1} = \emptyset$ (1)

By (1) and the definition of projection,

$\forall k < i, \sigma|_k = (\sigma \uplus \sigma')|_k$ (2)

By Lemma D.2,

$\sigma|_k, k \vDash \varphi$ iff $\sigma, k \vDash \varphi$ (3)

By Lemma D.2,

$\sigma \uplus \sigma'|_k, k \vDash \varphi$ iff $\sigma \uplus \sigma', k \vDash \varphi$ (4)

By (2),(3) and (4),

$\sigma, k \vDash \varphi$ iff $\sigma \uplus \sigma', k \vDash \varphi$ (5)

By (5),

$\sigma, i \vDash \diamondminus_{st}\, \varphi$ iff $\sigma \uplus \sigma', i \vDash \diamondminus_{st}\, \varphi$ (6)

□

$\boxed{\Sigma;\Gamma \vdash \varphi_f^* \ \mathsf{sat}}$ well-formed constraints on time points

$$\frac{}{\Sigma;\Gamma \vdash \top\ \mathsf{sat}} \qquad \frac{}{\Sigma;\Gamma \vdash \bot\ \mathsf{sat}} \qquad \frac{}{\Sigma;\Gamma \vdash A^p\ \mathsf{sat}} \qquad \frac{}{\Sigma;\Gamma \vdash \neg A^p\ \mathsf{sat}}$$

$$\frac{\Sigma;\Gamma \vdash \varphi_{f1}^*(p)\ \mathsf{sat} \qquad \Sigma;\Gamma \vdash \varphi_{f2}^*(p)\ \mathsf{sat}}{\Sigma;\Gamma \vdash \varphi_{f1}^*(p) \wedge \varphi_{f2}^*(p)\ \mathsf{sat}} \qquad \frac{\Sigma;\Gamma \vdash \varphi_f^*(p)\ \mathsf{sat}}{\Sigma;\Gamma \vdash \varphi_f^*(p) \vee \varphi\ \mathsf{sat}} \qquad \frac{\Sigma;\Gamma \vdash \varphi_f^+(p)\ \mathsf{sat}}{\Sigma;\Gamma \vdash \exists x.K^p(x) \wedge \varphi_f^+(p)\ \mathsf{sat}}$$

$$\frac{\Sigma;\Gamma \vdash \varphi_f^-(p)\ \mathsf{sat}}{\Sigma;\Gamma \vdash \forall x.\varphi_f^-(p)\ \mathsf{sat}} \qquad \frac{x;\cdot \vdash \varphi_f^*(p)\ \mathsf{sat}}{\cdot;\cdot \vdash {\downarrow}x.\varphi_f^*(p)\ \mathsf{sat}} \qquad \frac{\Sigma,y,x;\Gamma,y \le x \vdash \varphi_f^*(p)\ \mathsf{sat}}{\Sigma,y;\Gamma \vdash {\downarrow}x.\varphi_f^*(p)\ \mathsf{sat}} \qquad \frac{\Sigma;\Gamma \vdash \varphi_f^*(p)\ \mathsf{sat}}{\Sigma;\Gamma \vdash \Diamond\varphi_f^*(p)\ \mathsf{sat}}$$

$$\frac{\Sigma;\Gamma \vdash \varphi_f^*(p)\ \mathsf{sat}}{\Sigma;\Gamma \vdash \Box\varphi_f^*(p)\ \mathsf{sat}} \qquad \frac{\Sigma;\Gamma \vdash \varphi_{f1}^*(p)\ \mathsf{sat} \qquad \Sigma;\Gamma \vdash \varphi_{f2}^*(p)\ \mathsf{sat}}{\Sigma;\Gamma \vdash \varphi_{f1}^*(p)\,\mathcal{U}\,\varphi_{f2}^*(p)\ \mathsf{sat}}$$

$$\frac{\Sigma,y,x;\Gamma,y \le x, c(x) \vdash \varphi_f^*(p)\ \mathsf{sat} \qquad \Sigma,y;\Gamma \vdash \exists x.y \le x \wedge c(x)}{\Sigma,y;\Gamma \vdash \Diamond{\downarrow}x.\,c(x) \wedge \varphi_f^*(p)\ \mathsf{sat}}$$

$$\frac{\Sigma,y;\Gamma \vdash \varphi_{f1}^*(p)\ \mathsf{sat} \qquad \Sigma,y,x;\Gamma,y \le x, c(x) \vdash \varphi_{f2}^*(p)\ \mathsf{sat} \qquad \Sigma,y;\Gamma \vdash \exists x.y \le x \wedge c(x)}{\Sigma,y;\Gamma \vdash \varphi_{f1}^*\,\mathcal{U}\,({\downarrow}x.\,c(x) \wedge \varphi_{f2}^*(p))}$$

Figure 11: Rules for Checking Conditions for Time Points

#### D.2.3 Lemmas and Definitions for $\varphi_c^-$ and $\varphi_f^*$

**Satisfiability of Constraints on Time Points** Formulas such as $\varphi_f^+(p)$ and $\varphi_f^-(p)$ can be used to encode an agent $p$'s obligations. One can use the freeze operator to express a time bound on when $p$ has to finish his obligation. For such obligations to be feasible for $p$, constraints expressing these bounds should be satisfiable. Judgment $\Sigma;\Gamma \vdash \varphi_f^*\ \mathsf{sat}$ states that all the constraints on time points in $\varphi_f^*$ are satisfiable. $\Sigma$ contain all the free time variables in $\varphi_f^*$ and $\Gamma$; and $\Gamma$ is the context containing assumptions about various time points. The rules are shown in Figure 11.

**Lemma D.7** (Substitution for Conditions)**.**
*If* $\Sigma;\Gamma \vdash \varphi_f^*(p)\ \mathsf{sat}$ *, and* $\mathrm{dom}(\delta) \cap \Sigma = \emptyset$*, then* $\Sigma;\Gamma \vdash (\delta\varphi_f^*(p))\ \mathsf{sat}$

**Proof** (sketch): By induction on the derivation $\Sigma;\Gamma \vdash \varphi_f^*(p)$. □

**Lemma D.8** (Time Points Substitution for Conditions)**.**
*If* $\Sigma_1,\Sigma_2;\Gamma \vdash \varphi_f^*(p)\ \mathsf{sat}$ *, and* $\mathrm{dom}(\delta) = \Sigma_1$*, then* $\Sigma_2;\delta\Gamma \vdash (\delta\varphi_f^*(p))\ \mathsf{sat}$

**Proof** (sketch): By induction on the derivation $\Sigma_1,\Sigma_2;\Gamma \vdash \varphi_f^*(p)$. □

**Relevant Actions in $\varphi_c^-$ and $\varphi_f^*$** To precisely state what kind of actions and inaction an agent $p$ need to plan for to fulfill her responsibilities, we define a function $\mathcal{A}_c(\varphi_c^-(p))$ to extract relevant actions in $\varphi_c^-(p)$, and $\mathcal{A}_f(\varphi_f^*(p))$ to extract relevant actions in $\varphi_f^*(p)$. Detailed rules are defined in Figure 12.

**Lemma D.9** (Substitution for Actions)**.**

1. $\mathcal{A}_c(\varphi_c^-(p)\{t/x\}) = (\mathcal{A}_c(\varphi_c^-(p)))\{t/x\}$
2. $\mathcal{A}_c(\varphi_f^*(p)\{t/x\}) = (\mathcal{A}_c(\varphi_f^*(p)))\{t/x\}$

$\boxed{\mathcal{A}_c(\varphi_c^-)}$ the set of inactions involved in causing $\varphi_c^-$ to be false.

$$\mathcal{A}_c(A^p) = \{A^p\} \qquad \mathcal{A}_c(\varphi_{c1}^- \vee \varphi_{c2}^-) = \mathcal{A}_c(\varphi_{c1}^-) \cup \mathcal{A}_c(\varphi_{c2}^-) \qquad \mathcal{A}_c(\varphi_c^- \wedge \varphi) = \mathcal{A}_c(\varphi_c^-)$$

$$\mathcal{A}_c(\exists x.\varphi_c^-) = \mathcal{A}_c(\{a/x\}\varphi_c^-) \ a \text{ is fresh}$$

$\boxed{\mathcal{A}_f(\varphi_f^*)}$ the set of actions (inactions) involved in causing $\varphi_f^*$ to be true.

$$\mathcal{A}_f(A^p) = \{A^p\} \qquad \mathcal{A}_f(\neg A^p) = \{A^p\} \qquad \mathcal{A}_f(\varphi_f^* \vee \varphi) = \mathcal{A}_f(\varphi_f^*) \qquad \mathcal{A}_f(\varphi_{f1}^* \wedge \varphi_{f2}^*) = \mathcal{A}_f(\varphi_{f1}^*) \cup \mathcal{A}_f(\varphi_{f2}^*)$$

$$\mathcal{A}_f(U_{op}\ \varphi_f^*) = \mathcal{A}_f(\varphi_f^*) \qquad \mathcal{A}_f(\exists x.\varphi_f^*) = \mathcal{A}_f(\{a/x\}\varphi_f^*) \ a \text{ is fresh} \qquad \mathcal{A}_f(\varphi_f^*) = \emptyset \text{ for all other cases}$$

Figure 12: Function for Extracting Actions

$\boxed{\widehat{\sigma}, i \nVdash \varphi_c^-}$

$$\begin{array}{lll}
\widehat{\sigma}, i \nVdash \bot & & \text{always} \\
\widehat{\sigma}, i \nVdash A^p & \text{iff} & A^p \in \neg a(i) \\
\widehat{\sigma}, i \nVdash \varphi_c^- \wedge \varphi & \text{iff} & \widehat{\sigma}, i \nVdash \varphi_c^- \\
\widehat{\sigma}, i \nVdash \varphi_{c1}^- \vee \varphi_{c2}^- & \text{iff} & \widehat{\sigma}, i \nVdash \varphi_{c1}^- \text{ and } \widehat{\sigma}, i \nVdash \varphi_{c2}^- \\
\widehat{\sigma}, i \nVdash \exists x.\varphi_c^- & \text{iff} & \text{for all } t, \widehat{\sigma}, i \nVdash \{t/x\}\varphi_c^-
\end{array}$$

$\boxed{\widehat{\sigma}, i \Vdash \varphi_f^*}$

$$\begin{array}{lll}
\widehat{\sigma}, i \Vdash \top & & \text{always} \\
\widehat{\sigma}, i \Vdash \bot & & \text{never} \\
\widehat{\sigma}, i \Vdash A^p & \text{iff} & A^p \in a(i) \\
\widehat{\sigma}, i \Vdash \neg A^p & \text{iff} & A^p \in \neg a(i) \\
\widehat{\sigma}, i \Vdash \varphi_f^* \vee \varphi & \text{iff} & \widehat{\sigma}, i \Vdash \varphi_f^* \\
\widehat{\sigma}, i \Vdash \exists x.K^p(x) \wedge \varphi_f^+ & \text{iff} & \text{exists } t \text{ such that } \widehat{\sigma}, i \Vdash \{t/x\}\varphi_f^+ \\
 & & \text{and } \mathit{Tr}(\widehat{\sigma}), i \vDash K^p(t) \\
\widehat{\sigma}, i \Vdash \forall x.\varphi_f^- & \text{iff} & \text{forall}, t\ \widehat{\sigma}, i \Vdash \{t/x\}\varphi_f^- \\
\widehat{\sigma}, i \Vdash \downarrow x.\varphi_f^- & \text{iff} & \widehat{\sigma}, i \Vdash \{\tau(i)/x\}\varphi_f^- \\
\widehat{\sigma}, i \Vdash \Diamond\varphi_f^* & \text{iff} & \text{exists } j \text{ such that } j \geq i \text{ and } \widehat{\sigma}, j \Vdash \varphi_f^* \\
\widehat{\sigma}, i \Vdash \Box\varphi_f^* & \text{iff} & \text{for all } j \text{ such that } j \geq i \text{ and } \widehat{\sigma}, j \Vdash \varphi_f^* \\
\widehat{\sigma}, i \Vdash \varphi_{f1}^* \,\mathcal{U}\, \varphi_{f2}^* & \text{iff} & \text{exists} j, j \geq i \text{and } \widehat{\sigma}, j \Vdash \varphi_{f2}^* \text{ and } \forall i \geq k < j, \widehat{\sigma}, k \Vdash \varphi_{f1}^* \\
\widehat{\sigma}, i \Vdash c(t) \wedge \varphi_f^+ & \text{iff} & \widehat{\sigma}, i \Vdash \varphi_f^+ \text{and } \mathit{Tr}(\widehat{\sigma}), i \vDash c(t) \\
\widehat{\sigma}, i \Vdash \forall x.\varphi_f^- & \text{iff} & \text{for all}, t\ \widehat{\sigma}, i \Vdash \{t/x\}\varphi_f^-
\end{array}$$

Figure 13: Non-standard Semantics for Planned Traces

**Semantics for Planned Traces** Because the conjunction in $\varphi_c^-$ and the disjunction in $\varphi_f^*$ allows arbitrary formula as one of the subformulas, we define a non-standard semantics (Figure 13).

**Lemma D.10** (Soundness of Planned trace semantics)**.**

- *if* $\widehat{\sigma} \Vdash \varphi_f^*$*, and* $wf(\widehat{\sigma})$*, then* $\mathit{Tr}(\widehat{\sigma}) \vDash \varphi_f^*$
- *if* $\widehat{\sigma} \nVdash \varphi_c^-$*, and* $wf(\widehat{\sigma})$*, then* $\mathit{Tr}(\widehat{\sigma}) \nvDash \varphi_c^-$

*Proof.* By induction on the structure of the formula. □

We further define non-standard semantics for the three main forms of responsibilities.

$$
\begin{array}{lll}
\widehat{\sigma}, i \Vdash \forall \vec{x}.\varphi_c^- \supset \varphi_{past} & \text{iff} & \text{for all } \vec{t} \text{ for } \vec{x},\ \text{either } \widehat{\sigma}, i \nVdash \varphi_c^-\{\vec{t}/\vec{x}\} \text{ or } Tr(\widehat{\sigma}), i \vDash \varphi_{past}\{\vec{t}/\vec{x}\} \\
\widehat{\sigma}, i \Vdash \forall \vec{x}.\varphi_{past} \supset \varphi_f^+ & \text{iff} & \text{for all } \vec{t} \text{ for } \vec{x},\ \text{such that } Tr(\widehat{\sigma}), i \vDash \varphi_{past}\{\vec{t}/\vec{x}\} \text{ implies } \widehat{\sigma}, i \Vdash \varphi_f^+\{\vec{t}/\vec{x}\} \\
\widehat{\sigma}, i \Vdash \forall \vec{x}.\varphi_{past} \supset \varphi_f^- & \text{iff} & \text{for all } \vec{t} \text{ for } \vec{x}, \widehat{\sigma}, i \Vdash \varphi_f^-\{\vec{t}/\vec{x}\}
\end{array}
$$

**Lemma D.11** (Soundness of non-standard Semantics)**.** *if* $\widehat{\sigma} \Vdash \varphi_i$ *as defined above then* $Tr(\widehat{\sigma}) \vDash \varphi_i$

*Proof.* By Lemma D.10. □

### Key Lemmas About $\varphi_c^-$ and $\varphi_f^*$

**Lemma D.12** (Monotonicity of $\varphi_c^-$)**.**
*Given any* $\sigma$, $i$ *and* $\sigma'$ *such that* $\sigma \uplus \sigma'$ *is well-defined and* $\sigma, i \nVdash \varphi_c^-(p)$ *implies* $\sigma \uplus \sigma', i \nVdash \varphi_c^-(p)$

**Proof** (sketch): By induction on the structure of $\varphi_c^-$. □

**Lemma D.13** (Feasibility of $\varphi_c^-(p)$)**.**
*For all* $i$*, there exists a set of inactions* $NAS$ *such that*

*I.* $p$ *is the performer of all actions in* $NAS$*,*

*II. for all* $P(\vec{s}) \in NAS$ *there exists a* $P(\vec{w}) \in \mathcal{A}_c(\varphi_c^-(p))$ *such that there exists a substitution* $\delta$*, and* $P(\delta(\vec{s}) = P(\delta(\vec{w}))$*.*

*III. for all* $\widehat{\sigma}$*,* $\neg a(i) \supseteq NAS$*,* $\widehat{\sigma}, i \nVdash \varphi_c^-$

*Proof.* By induction on the structure of $\varphi_c^-$.

Case $\varphi_c^- = \bot$
$NAS = \{\}$

Case $\varphi_c^- = A^p$
$NAS = \{A^p\}$
Given $\widehat{\sigma}$, such that $\neg a(i) \supseteq \{A^p\}$ ,
By Definition of $\widehat{\sigma}, i \nVdash \varphi_c^-$,
$\widehat{\sigma}, i \nVdash A^p$

Case $\varphi_c^- = \varphi_{c1}^-(p) \wedge \varphi$
By I.H. on $\varphi_{c1}^-(p)$, there exists a set of inactions $NAS_1$, such that
$p$ is the performer of all actions in $NAS_1$ (1)
for all $P(\vec{s}) \in NAS_1$ there exists a $P(\vec{w}) \in \mathcal{A}_c(\varphi_{c1}^-(p))$ such that
there exists a substitution $\delta$, and $P(\delta\vec{s}) = P(\delta\vec{w})$. (2)
for all $\widehat{\sigma}_1, \neg a(i) \supseteq NAS$, $\widehat{\sigma}_1, i \nVdash \varphi_{c1}^-(p)$ (3)
let $NAS = NAS_1$,
By (1), $I$ holds
By Definition of $\mathcal{A}_c(\varphi_c^-)$,
$\mathcal{A}_c(\varphi_{c1}^-(p) \wedge \varphi) = \mathcal{A}_c(\varphi_{c1}^-(p))$ (4)
By (4) and (2), $II$ holds
By Given any $\widehat{\sigma}$, such that $a(i) \supseteq NAS$
By (3),
$\widehat{\sigma}, i \nVdash \varphi_{c1}^-(p)$ (5)
By Definition of $\nVdash$ and (5),
$\widehat{\sigma}, i \nVdash \varphi_{c1}^-(p) \wedge \varphi$ (6)

Case $\varphi_c^- = \varphi_{c1}^-(p) \wedge \varphi_{c2}^-(p)$

By I.H. on $\varphi_{c1}^-(p)$, there exists a set of inactions $NAS_1$, such that

$p$ is the performer of all actions in $NAS_1$ (1)

for all $P(\vec{s}) \in NAS_1$ there exists a $P(\vec{w}) \in \mathcal{A}_c(\varphi_{c1}^-(p))$ such that there exists a substitution $\delta$, and $P(\delta\vec{s}) = P(\delta\vec{w})$. (2)

for all $\widehat{\sigma}_1, \neg a_1(i) \supseteq NAS,\ \widehat{\sigma}_1, i \nVdash \varphi_{c1}^-(p)$ (3)

By I.H. on $\varphi_{c2}^-(p)$, there exists a set of inactions $NAS_1$, such that

$p$ is the performer of all actions in $NAS_2$ (4)

for all $P(\vec{s}) \in NAS_2$ there exists a $P(\vec{w}) \in \mathcal{A}_c(\varphi_{c2}^-(p))$ such that there exists a substitution $\delta$, and $P(\delta\vec{s}) = P(\delta\vec{w})$. (5)

for all $\widehat{\sigma}_2, \neg a_2(i) \supseteq NAS,\ \widehat{\sigma}_2, i \nVdash \varphi_{c2}^-(p)$ (6)

let $NAS = NAS_1 \cup NAS_2$,

By (1) and (4), $I$ holds

By Definition of $\mathcal{A}_c(\varphi_c^-)$,

$$\mathcal{A}_c(\varphi_{c1}^-(p) \wedge \varphi_{c2}^-(p)) = \mathcal{A}_c(\varphi_{c1}^-(p)) \cup \mathcal{A}_c(\varphi_{c2}^-(p)) \quad (7)$$

By (7), (2) and (5), $II$ holds

By Given any $\widehat{\sigma}$, such that $a(i) \supseteq NAS$

By (3),

$$\widehat{\sigma}, i \nVdash \varphi_{c1}^-(p) \quad (8)$$

By (6),

$$\widehat{\sigma}, i \nVdash \varphi_{c2}^-(p) \quad (9)$$

By Definition of $\nVdash$,(8) and (9),

$$\widehat{\sigma}, i \nVdash \varphi_{c1}^-(p) \wedge \varphi_{c2}^-(p) \quad (10)$$

Case $\varphi_c^- = \exists x.\varphi_{c1}^-(p)$

For all $t$,

By I.H. on $\varphi_{c1}^-(p)\{t/x\}$, there exists a set of inactions $NAS_t$, such that

$p$ is the performer of all actions in $NAS_t$ (1)

for all $P(\vec{s}) \in NAS_t$ there exists a $P(\vec{w}) \in \mathcal{A}_c(\varphi_{c1}^-(p)\{t/x\})$ such that there exists a substitution $\delta$, and $P(\delta\vec{s}) = P(\delta\vec{w})$. (2)

for all $\widehat{\sigma}_1, \neg a(i) \supseteq NAS_t,\ \widehat{\sigma}_1, i \nVdash \varphi_{c1}^-(p)\{t/x\}$ (3)

let $NAS = \bigcup_t NAS_t$,

By (1), $I$ holds

By Definition of $\mathcal{A}_c(\varphi_c^-)$,

$$\mathcal{A}_c(\exists x.\varphi_{c1}^-(p)) = \mathcal{A}_c(\varphi_{c1}^-(p)) \quad (4)$$

By Lemma D.9,

$$\mathcal{A}_c(\varphi_{c1}^-(p)\{t/x\}) = (\mathcal{A}_c(\varphi_{c1}^-(p)))\{t/x\} \quad (5)$$

By (4), (2) and (5), given any $P(\vec{s}) \in NAS$

exists $P(\vec{w_1}) \in (\mathcal{A}_c(\varphi_{c1}^-(p)))$ such that $\delta_1 = (\delta, t/x)$ and$P(\delta\vec{s}) = P(\delta_1\vec{w_1}) = P(\delta\vec{w})$ (6)

By Given any $\widehat{\sigma}$, such that $a(i) \supseteq NAS \supseteq NAS_t$

By (3),

$$\widehat{\sigma}, i \nVdash \varphi_{c1}^-(p)\{t/x\} \quad (7)$$

By Definition of $\nVdash$ and (7),

$$\widehat{\sigma}, i \nVdash \exists x.\varphi_{c1}^-(p) \quad (8)$$

□

**Lemma D.14** (Feasibility of Conditions)**.** *Given a condition $K^p(\vec{x})$, for all $i$, exists $\vec{t}$ such that for all $\widehat{\sigma}$, $\kappa \supseteq \kappa^p$, $\widehat{\sigma}, i \vdash K^p(\vec{t})$.*

**Proof** (sketch): By induction on the structure of $K^p(\vec{x})$ □

We further assume that solutions for the constraint $c(x)$ on time points are multiples of $\imath$.

**Lemma D.15** (Monotonicity of $\varphi_f^+$).
*If $\widehat{\sigma}, i \Vdash \varphi_f^+(p)$, given $\widehat{\sigma}'$, such that $\widehat{\sigma} \uplus \widehat{\sigma}'$ is defined, then $\widehat{\sigma} \uplus \widehat{\sigma}', i \Vdash \varphi_f^+(p)$*

**Proof** (sketch): By induction on the structure of $\varphi_f^+(p)$. □

**Lemma D.16** (Feasibility of $\varphi_f^+(p)$).
*If $\Sigma; \Gamma \vdash \varphi_f^+(p)$ $\mathsf{sat}$ then $\forall t_0$, $\forall i$, $\forall \delta$ such that*

1. $\forall t \in \mathit{range}(\delta)$, $\exists n \geq 0, t = t_0 + n \cdot \imath$
2. $\delta(x) \leq \delta(y)$ *if $x$ appears before $y$ in $\Sigma$*
3. $\delta(z) = t_0 + i \cdot \imath$ *where $\Sigma = \Sigma', z$*
4. $\vDash \delta\Gamma$

*there exists a finite action map $a'$, and $\tau'$ such that $\mathit{start}(\tau') = t_0$*

I. $\forall j \in \mathrm{dom}(a')$, $j \geq i$, $\forall j \in \mathrm{dom}(\tau)$, $j \geq i$
II. *$p$ is the performer of all actions in the range of $a'$,*
III. *for all $P(\vec{s}) \in \mathit{range}(a')$ there exists a $P(\vec{w}) \in \mathcal{A}_f(\varphi_f^+(p))$ such that there exists a substitution $\delta^0$, and $P(\delta^0(\vec{s}) = P(\delta^0(\vec{w}))$.*
IV. *for all $\widehat{\sigma}$, $\widehat{\sigma} \supseteq (\kappa^p, \emptyset, \emptyset, a', \emptyset, \tau', \emptyset)$, $\widehat{\sigma}, i \Vdash \delta\varphi_f^+(p)$*

*Proof.* By induction on the structure of $\varphi_f^+(p)$.
Most cases are straightforward, and we only give $a'$ and $\tau'$. We focus on a few key cases involving the freeze operator.

Case $\varphi_f^+(p) = \top$, $a' = \emptyset$ and $\tau' = \emptyset$

Case $\varphi_f^+(p) = A^p$
$a'(i) = \{\delta A^p\}$, and $\tau' = \{i \mapsto t_0 + i \cdot \imath\}$

Case $\varphi_f^+(p) = \varphi_{f1}^+(p) \wedge \varphi_{f2}^+(p)$
By I.H. on $\varphi_{f1}^+(p)$, there exists $\tau_1$ and $a_1$ that satisfy all the conditions
By I.H. on $\varphi_{f2}^+(p)$, there exists $\tau_2$ and $a_2$ that satisfy all the conditions
$\tau' = \tau_1 \sqcup \tau_2$, and $a' = a_1 \cup a_2$,

Case $\varphi_f^+(p) = \varphi_{f1}^+(p) \vee \varphi$
By I.H. on $\varphi_{f1}^+(p)$, there exists $\tau_1$ and $a_1$ that satisfy all the conditions
$\tau' = \tau_1$, and $a' = a_1$,

Case $\varphi_f^+(p) = \exists x.K^p(x) \wedge \varphi_{f1}^+(p)$
By A,ssumptions
$$\frac{\Sigma; \Gamma \vdash \varphi_{f1}^+(p) \ \mathsf{sat}}{\Sigma; \Gamma \vdash \exists x.K^p(x) \wedge \varphi_{f1}^+(p) \ \mathsf{sat}} \quad (1)$$
Given $t_0$, $i$, $\delta$ such that,
$\forall t \in \mathrm{range}(\delta), \exists n \geq 0, t = t_0 + n \cdot \imath$ (2)
$\delta(x) \leq \delta(y)$ if $x$ appears before $y$ in $\Sigma$ (3)
$\delta(z) = t_0 + i \cdot \imath$ where $\Sigma = \Sigma', z$ (4)
$\vDash \delta\Gamma$ (5)
By Lemma D.14,
exists $t$ such that for all $\widehat{\sigma}$ such that $\kappa \supseteq \kappa^p, \widehat{\sigma}, i \vdash \delta K^p(t)$ (6)
By Lemma D.7 and (1),
$\Sigma; \Gamma \vdash \varphi_{f1}^*(p)\{t/x\}$ $\mathsf{sat}$ (7)

By I.H. on $\varphi^*_{f1}(p)\{t/x\}$, and (1) to (5),

there exists an action map $a_1$, and $\tau_1$ such that $\mathit{start}(\tau_1) = t_0$

$$\forall j \in \mathrm{dom}(a_1), j \geq i, \text{ and } \forall j \in \mathrm{dom}(\tau_1), j \geq i \tag{8}$$

$p$ is the performer of all actions in the range of $a_1$ (9)

for all $P(\vec{s}) \in \mathrm{range}(a_1)$ there exists a $P(\vec{w}) \in \mathcal{A}_f(\varphi^+_f(p)\{t/x\})$ such that there exists a substitution $\delta^0$, and $P(\delta^0\vec{s}) = P(\delta^0(\vec{w}))$. (10)

$$\forall \widehat{\sigma}, \widehat{\sigma} \supseteq (\kappa^p, \emptyset, \emptyset, a_1, \emptyset, \tau_1, \emptyset), \widehat{\sigma}, i \Vdash \delta\varphi^+_{f1}(p)\{t/x\} \tag{11}$$

let $\tau' = \tau_1$, $a' = a_1$

By (8), $I$ holds

By (9), $II$ holds

By Definition of $\mathcal{A}_f(\varphi^+_f(p))$,

$$\mathcal{A}_f(\exists x.K^p(x) \wedge \varphi^+_{f1}(p)) = \mathcal{A}_f(\varphi^+_{f1}(p)) \tag{12}$$

By Lemma D.9,

$$\mathcal{A}_f(\varphi^+_{f1}(p)\{t/x\}) = (\mathcal{A}_f(\varphi^+_{f1}(p)))\{t/x\} \tag{13}$$

By (12), (10) and (13), given any $P(\vec{s}) \in \mathrm{range}(a)$

$$\text{exists } P(\vec{w_1}) \in (\mathcal{A}_f(\varphi^+_{f1}(p))) \text{ such that } \delta_1 = (\delta^0, t/x) \text{ and } P(\delta^0\vec{s}) = P(\delta_1\vec{w_1}) = P(\delta^0\vec{w}) \tag{14}$$

Give $\widehat{\sigma}$, $\widehat{\sigma} \supseteq (\kappa^p, \emptyset, \emptyset, a', \emptyset, \tau', \emptyset)$

By (11),

$$\widehat{\sigma}, i \Vdash \delta\varphi^+_{f1}(p)\{t/x\} \tag{15}$$

By (15), (6),

$$\widehat{\sigma}, i \Vdash \delta(\exists x.K^p(x) \wedge \varphi^+_{f1}(p)) \tag{16}$$

Case $\varphi^+_f(p) = \Diamond\varphi^+_{f1}(p)$

By I.H. on $\varphi^+_{f1}(p)$, there exists $a_1$ and $\tau_1$ that satisfy all the conditions

$a' = a_1$ and $\tau' = \tau_1$,

Case $\varphi^+_f(p) = \varphi^+_{f2}(p)\,\mathcal{U}\,\varphi^+_{f1}(p)$

By I.H. on $\varphi^+_{f1}(p)$, there exists $a_1$ and $\tau_1$ that satisfy all the conditions

$\kappa = \kappa$, $a' = a_1$ and $\tau' = \tau_1$,

Case $\varphi^+_f(p) = \Box\varphi^+_{f1}(p)$

By A,ssumptions

$$\frac{\Sigma; \Gamma \vdash \varphi^+_{f1}(p)\ \mathsf{sat}}{\Sigma; \Gamma \vdash \Box\varphi^+_{f1}(p)\ \mathsf{sat}} \tag{1}$$

Given $t_0$, $i$, $\delta$ such that

$$\forall t \in \mathrm{range}(\delta), \exists n \geq 0, t = t_0 + n \cdot \imath \tag{2}$$

$$\delta(x) \leq \delta(y) \text{ if } x \text{ appears before } y \text{ in } \Sigma \tag{3}$$

$$\delta(z) = \tau(i) \text{ where } \Sigma = \Sigma', z \tag{4}$$

$$\vDash \delta\Gamma \tag{5}$$

By Lemma D.8 and (1),

$$\cdot; \delta\Gamma \vdash \delta\varphi^+_{f1}(p)\ \mathsf{sat} \tag{6}$$

Given any $k$, $k \geq i$, by I.H. on $\delta\varphi^+_{f1}(p)$,

there exists an action map $a_k$, and $\tau_k$ such that $\mathit{start}(\tau_k) = t_0$

$$\forall j \in \mathrm{dom}(a_k), j \geq k, \text{ and } \forall j \in \mathrm{dom}(\tau_k), j \geq k \tag{7}$$

$p$ is the performer of all actions in the range of $a_1$ (8)

for all $P(\vec{s}) \in \mathrm{range}(a_k)$ there exists a $P(\vec{w}) \in \mathcal{A}_f(\delta\varphi^+_f(p))$ such that there exists a substitution $\delta^0$, and $P(\delta^0\vec{s}) = P(\delta^0(\vec{w}))$. (9)

$$\forall \widehat{\sigma}, \widehat{\sigma} \supseteq (\kappa^p, \emptyset, \emptyset, a_k, \emptyset, \tau_k, \emptyset), \widehat{\sigma}, k \Vdash \delta\varphi^+_f(p) \tag{10}$$

let $a' = \bigcup_{k=1}^{\infty} a_k$, $\tau' = \bigsqcup_{k=1}^{\infty} \tau_k$

By (7), $I$ holds

By (8), $II$ holds

By (9) and Lemma D.9, $III$ holds
Give $\widehat{\sigma}$, $\widehat{\sigma} \supseteq (\kappa^p, \emptyset, \emptyset, a', \emptyset, \tau', \emptyset)$
By (10) and the definition of $\Vdash$,

$$\widehat{\sigma}, i \Vdash \delta\Box\varphi^+_{f1}(p) \tag{11}$$

Case $\varphi^+_f(p) = \downarrow x.\varphi^+_{f1}(p)$
By Assumptions,

$$\frac{x; \cdot \vdash \varphi^+_{f1}(p) \ \mathsf{sat}}{\cdot; \cdot \vdash \downarrow x.\varphi^+_{f1}(p) \ \mathsf{sat}} \tag{1}$$

Given $t_0$, $i$ (because $\Sigma$ is empty, $\delta$ is empty also),
By I.H. on $\varphi^+_{f1}(p)$, let $\delta = t_0 + i \cdot \imath / x$
there exists an action map $a_1$, and $\tau_1$ such that $start(\tau_1) = t_0$

$\forall j \in \mathrm{dom}(a_1), j \geq i,$ and $\forall j \in \mathrm{dom}(\tau_1), j \geq i$ (2)
$p$ is the performer of all actions in the range of $a_1$ (3)
for all $P(\vec{s}) \in \mathrm{range}(a_1)$ there exists a $P(\vec{w}) \in \mathcal{A}_f(\varphi^+_f(p))$ such that
there exists a substitution $\delta^0$, and $P(\delta^0\vec{s}) = P(\delta^0(\vec{w}))$. (4)
$\forall \widehat{\sigma}, \widehat{\sigma} \supseteq (\kappa^p, \emptyset, \emptyset, a_1, \emptyset, \tau_1, \emptyset), \widehat{\sigma}, i \Vdash \delta\varphi^+_f(p)$ (5)

let $\tau' = \tau_1$ and $a' = a_1$,
By (2), $I$ holds
By (3), $II$ holds
By (4), $III$ holds
Give $\widehat{\sigma}$, $\widehat{\sigma} \supseteq (\kappa^p, \emptyset, \emptyset, a', \emptyset, \tau', \emptyset)$
By (5), $\delta = \tau'(i)/x$ and the definition of $\Vdash$,

$$\widehat{\sigma}, i \Vdash \downarrow x.\varphi^+_{f1}(p) \tag{6}$$

Case $\varphi^+_f(p) = \downarrow x.\varphi^+_{f1}(p)$

$$\frac{\Sigma, y, x; \Gamma, y \leq x \vdash \varphi^+_f(p) \ \mathsf{sat}}{\Sigma, y; \Gamma \vdash \downarrow x.\varphi^+_f(p) \ \mathsf{sat}} \tag{1}$$

Given $t_0$, $i$, $\delta$ such that,

$\forall t \in \mathrm{range}(\delta), \exists n \geq 0, t = t_0 + n \cdot \imath$ (2)
$\delta(x') \leq \delta(y')$ if $x'$ appears before $y'$ in $\Sigma, y$ (3)
$\delta(z) = t_0$ where $\Sigma = \Sigma', z$ (4)
$\vDash \delta\Gamma$ (5)

let $\delta_1 = \delta, t_0 + i \cdot \imath / x$,
By (2),

$\forall t \in \mathrm{range}(\delta_1), \exists n \geq 0, t = t_0 + n \cdot \imath$ (6)

By (3), (4),

$\delta(y) = \delta_1(x),$ therefore $\delta_1(x') \leq \delta_1(y')$ if $x$ appears before $y$ in $\Sigma, y, x$ (7)

By (5), and $\delta_1(y) = \delta_1(x) = t_0 + i \cdot \imath$

$\vdash \delta_1(\Gamma, y \leq x)$ (8)

By I.H. on $\varphi^+_{f1}(p)$, and (6) to (8),
there exists an action map $a_1$, and $\tau_1$ such that $start(\tau_1) = t_0$

$\forall j \in \mathrm{dom}(a_1), j \geq i,$ and $\forall j \in \mathrm{dom}(\tau_1), j \geq i$ (9)
$p$ is the performer of all actions in the range of $a_1$ (10)
for all $P(\vec{s}) \in \mathrm{range}(a_1)$ there exists a $P(\vec{w}) \in \mathcal{A}_f(\varphi^+_f(p)\{t/x\})$ such that
there exists a substitution $\delta^0$, and $P(\delta^0\vec{s}) = P(\delta^0(\vec{w}))$. (11)
$\forall \widehat{\sigma}, \widehat{\sigma} \supseteq (\kappa^p, \emptyset, \emptyset, a_1, \emptyset, \tau_1, \emptyset), \widehat{\sigma}, i \Vdash \delta_1\varphi^+_f(p)$ (12)

let $\tau' = \tau_1$, $a' = a_1$
By (9), $I$ holds
By (10), $II$ holds
By (11), $III$ holds

Give $\widehat{\sigma}$, $\widehat{\sigma} \supseteq (\kappa, \emptyset, \emptyset, a', \emptyset, \tau', \emptyset)$
By (12), $\delta_1 = \delta, \tau'(i)/x$ and the definition of $\Vdash$,
$$\widehat{\sigma}, i \Vdash \delta{\downarrow}x.\varphi^+_{f1}(p) \tag{13}$$

Case $\varphi^+_f(p) = \Diamond{\downarrow}x.\, c(x) \wedge \varphi^+_f(p)$
$$\frac{\mathcal{E}_1 :: \Sigma, y, x; \Gamma, y \leq x, c(x) \vdash \varphi^+_f(p) \qquad \mathcal{E}_2 :: \Sigma, y; \Gamma \vdash \exists x.y \leq x \wedge c(x)}{\Sigma, y; \Gamma \vdash \Diamond{\downarrow}x.\, c(x) \wedge \varphi^+_f(p)} \tag{1}$$
Given $t_0$, $i$, $\delta$ such that,
$$\forall t \in \text{range}(\delta), \exists n \geq 0, t = t_0 + n \cdot \imath \tag{2}$$
$$\delta(x') \leq \delta(y') \text{ if } x' \text{ appears before } y' \text{ in } \Sigma, y \tag{3}$$
$$\delta(y) = t_0 + i \cdot \imath \tag{4}$$
$$\vDash \delta\Gamma \tag{5}$$
By $\mathcal{E}_2$ and (5),
$$\vDash \delta(\exists x.y \leq x \wedge c(x)) \tag{6}$$
$$\text{there exists } \tau_x \text{ such that } \vDash \delta(y) \leq \tau_x \text{ and } \vDash \delta c(\tau_x) \tag{7}$$
let $\delta_1 = \delta, \tau_x/x$,
By (7) and (3),
$$\delta_1(x') \leq \delta_1(y') \text{ if } x \text{ appears before } y \text{ in } \Sigma, y, x \tag{8}$$
By (5), (7),
$$\vDash \delta_1(\Gamma, y \leq x, c(x)) \tag{9}$$
By assumptions that $\tau_x$ is a multiple of $\imath$,
let $n = i + (\tau_x - \tau(i))/\imath$, let $\tau'_1$ be a mapping containing the only following mapping $\tau'(n) = \tau_x$,
By I.H. on $\varphi^+_{f1}(p)$, and (8) to (9),
there exists an action map $a_1$, and $\tau_1$ such that $\mathit{start}(\tau_1) = t_0$
$$\forall j \in \text{dom}(a_1), j \geq n, \text{ and } \forall j \in \text{dom}(\tau_1), j \geq n \tag{10}$$
$$p \text{ is the performer of all actions in the range of } a_1 \tag{11}$$
for all $P(\vec{s}) \in \text{range}(a_1)$ there exists a $P(\vec{w}) \in \mathcal{A}_f(\varphi^+_f(p)\{t/x\})$ such that
$$\text{there exists a substitution } \delta^0, \text{ and } P(\delta^0\vec{s}) = P(\delta^0(\vec{w})). \tag{12}$$
$$\forall \widehat{\sigma}, \widehat{\sigma} \supseteq (\kappa^p, \emptyset, \emptyset, a_1, \emptyset, \tau_1, \emptyset), \widehat{\sigma}, n \Vdash \delta_1\varphi^+_f(p) \tag{13}$$
let $\tau' = \tau'_1$, $a' = a_1$
By (10), $I$ holds
By (11), $II$ holds
By (12), $III$ holds
Give $\widehat{\sigma}$, $\widehat{\sigma} \supseteq (\kappa^p, \emptyset, \emptyset, a', \emptyset, \tau', \emptyset)$
By (13) and (7) $\delta_1 = \delta, \tau(n)/x$ and the definition of $\Vdash$,
$$\widehat{\sigma}, i \Vdash \delta\Diamond{\downarrow}x.c(x) \wedge \varphi^+_{f1}(p) \tag{14}$$

Case: $\varphi^+_{f1}\,\mathcal{U}\,({\downarrow}x.\, c(x) \wedge \varphi^+_{f2}(p))$
Given $t_0$, $i$, $\delta$,
By I.H. on $\varphi^+_{f2}(p)$, we can find a time point $\tau_x$, which maps to state $n$ such that
there exists $a_2$ and $\tau_2$ that satisfy all the conditions,
$$\text{and for all } \widehat{\sigma} \supseteq (\kappa^p, \emptyset, \emptyset, a_2, \emptyset, \tau_2, \emptyset),\ \widehat{\sigma}, n \Vdash \delta({\downarrow}x.c(x) \wedge \varphi^+_{f2}(p)) \tag{1}$$
By I.H. on $\delta\varphi^+_{f1}(p)$, there exists $a_k$ and $\tau_k$ for each $i \leq k < n$ that satisfy all the conditions
$$\text{and for all } \widehat{\sigma} \supseteq (\kappa^p, \emptyset, \emptyset, a_k, \emptyset, \tau_k, \emptyset),\ \widehat{\sigma}, k \Vdash \delta\varphi^+_{f1}(p) \tag{2}$$
let $a' = a_1 \cup \bigcup_{k=i}^{n} a_k$, $\tau' = \tau \sqcup \tau_2 \sqcup \bigsqcup_{k=i}^{n} \tau_k$
By (1) and (2),
$$\text{for all } \widehat{\sigma} \supseteq (\kappa^p, \emptyset, \emptyset, a', \emptyset, \tau', \emptyset),\ \widehat{\sigma}, i \Vdash \delta(\varphi^+_{f1}(p)\,\mathcal{U}\,{\downarrow}x.c(x) \wedge \varphi^+_{f2}(p)) \tag{3}$$

□

**Lemma D.17** (Monotonicity of $\varphi^-_f$).

*If $\widehat{\sigma}, i \Vdash \varphi_f^-(p)$, given $\widehat{\sigma}'$, such that $\widehat{\sigma} \uplus \widehat{\sigma}'$ is defined, then $\widehat{\sigma} \uplus \widehat{\sigma}', i \Vdash \varphi_f^+(p)$*

**Proof** (sketch): By induction on the structure of $\varphi_f^-$. □

**Lemma D.18** (Feasibility of $\varphi_f^-(p)$)**.**
*If $\Sigma; \Gamma \vdash \varphi_f^-(p)$ then $\forall \tau$, $\forall i$, $\forall \delta$ such that*

1. *$\forall t \in range(\delta)$, $\exists n \geq 0, t = t_0 + n \cdot \imath$*
2. *$\delta(x) \leq \delta(y)$ if $x$ appears before $y$ in $\Sigma$*
3. *$\delta(z) = \tau(i)$ where $\Sigma = \Sigma', z$*
4. *$\vDash \delta\Gamma$*

*there exists an action map $\neg a'$, and and $\tau'$ such that $start(\tau') = t_0$*

- I. *$\forall j \in \mathrm{dom}(\neg a')$, $j \geq i$, and $\forall j \in \mathrm{dom}(\tau')$, $j \geq i$*
- II. *$p$ is the performer of all actions in the range of $\neg a'$,*
- III. *for all $P(\vec{s}) \in range(\neg a')$ there exists a $P(\vec{w}) \in \mathcal{A}_f(\varphi_f^-(p))$ such that there exists a substitution $\delta^0$, and $P(\delta^0(\vec{s}) = P(\delta^0(\vec{w}))$.*
- IV. *for all $\widehat{\sigma}$, $\widehat{\sigma} \supseteq (\kappa^p, \emptyset, \emptyset, \emptyset, \neg a', \tau', \emptyset)$, $\widehat{\sigma}, i \Vdash \delta\varphi_f^-(p)$*

**Proof** (sketch): By induction on the structure of $\varphi_f^-(p)$. The proof is very similar to Lemma D.16. □

### D.2.4 Feasibility Theorems

To prove feasibility theorems, we first prove several stronger lemmas, which require stronger definitions for feasibility. The general structure of these feasibility definitions are as follows.

$$
\begin{array}{lcl}
H(\widehat{\sigma}, i, Sa, t_0) & = & (\widehat{\sigma}\mid_{i-1} = \emptyset) \wedge (\widehat{\sigma}\mid_i = \widehat{\sigma}) \wedge start(\tau) = t_0 \wedge \forall p \in Sa, (\widehat{\sigma}\mid_p^{\mathcal{A}} = \emptyset) \wedge \forall p \in Sa, \kappa \supseteq \kappa^p \\
G(a, \neg a, \tau, i, Sa, t_0) & = & \text{to be defined} \\
V(\widehat{\sigma}, i, \Phi) & = & \text{to be defined}
\end{array}
$$

$$
\begin{array}{lcl}
GF'(0, \Phi, Sa) & = & \forall \widehat{\sigma}_0, H(\widehat{\sigma}_0, 0, Sa, t_0) \supset \\
& & \quad \exists a'_0, \neg a'_0, \tau'_0, G(a'_0, \neg a'_0, \tau'_0, 0, Sa, t_0) \\
& & \qquad \wedge\ \widehat{\sigma}_0 \uplus (\emptyset, \emptyset, a'_0, \neg a'_0, \tau'_0, \emptyset) \text{ is well-defined} \\
& & \qquad \wedge\ \forall \widehat{\sigma}', \widehat{\sigma}'\mid_0 = \emptyset \wedge \widehat{\sigma}' \uplus \widehat{\sigma}_0 \uplus (\emptyset, \emptyset, a'_0, \neg a'_0, \tau'_0, \emptyset) \text{ is well-defined} \supset \\
& & \qquad V(\widehat{\sigma}' \uplus \widehat{\sigma}_0 \uplus (\emptyset, \emptyset, a'_0, \neg a'_0, \tau'_0, \emptyset), 0, \Phi) \\
& & \qquad \wedge\ [\ ]_{\widehat{\sigma}_0, a'_0, \neg a'_0, \tau'_0} \\
GF'(j, \Phi, Sa) & = & \\
\multicolumn{3}{l}{\quad GF'(j-1, \Phi, Sa)[\forall \widehat{\sigma}_j, H(\widehat{\sigma}_j, j, Sa, t_0) \supset} \\
& & \quad (\exists a'_j, \neg a'_j, \tau'_j, G(a'_j, \neg a'_j, \tau'_j, j, Sa, t_0) \\
& & \qquad \wedge\ \biguplus_{k=0}^{j} \widehat{\sigma}_k \uplus \biguplus_{k=0}^{j} (\emptyset, \emptyset, a'_k, \neg a'_k, \tau'_k, \emptyset) \text{ is well-defined} \\
& & \qquad \wedge\ \forall \widehat{\sigma}', \widehat{\sigma}'\mid_j = \emptyset \wedge \widehat{\sigma}' \uplus \biguplus_{k=0}^{j} \widehat{\sigma}_k \uplus \biguplus_{k=0}^{j} (\emptyset, \emptyset, a'_k, \neg a'_k, \tau'_k, \emptyset) \text{ is well-defined} \supset \\
& & \qquad V(\widehat{\sigma}' \uplus \biguplus_{k=0}^{j} \widehat{\sigma}_k \uplus \biguplus_{k=0}^{j} (\emptyset, \emptyset, a'_k, \neg a'_k, \tau'_k, \emptyset), j, \Phi) \\
& & \qquad \wedge\ [\ ]_{\widehat{\sigma}_0, a'_0, \neg a'_0, \tau'_0, \cdots \widehat{\sigma}_{j-1}, a'_{j-1}, \neg a'_{j-1}, \tau'_{j-1}} \\
& & ]_{\widehat{\sigma}_0, a'_0, \neg a'_0, \tau'_0, \cdots \widehat{\sigma}_{j-1}, a'_{j-1}, \neg a'_{j-1}, \tau'_{j-1}, \widehat{\sigma}_j,, a'_j, \neg a'_j, \tau'_j}
\end{array}
$$

The definition of $GF'(\widehat{\sigma}, j, \Phi)$ is very similar to the definition of $GF(\widehat{\sigma}, j, \Phi)$. We left abstract, the properties for the existentially quantified action map, inaction map and time stamp map; and the properties of the final trace. Each feasibility lemma will instantiate $G$ and $V$ so that the induction hypothesis is strong enough to prove the lemma.

In the special case when $Sa = \{p\}$, we simply write $GF'(j, \Phi, p)$.

**Lemma D.19** (Feasibility of (r1) in one state)**.**

*Given state $i$, $\widehat{\sigma}$ such that $\kappa \supseteq \kappa^p$, and $\forall j \geq i, \forall P \in a(j)$, $p$ is not the performer of $P$, there exists a set of inactions $NAS$ such that*

*I. $p$ is the performer of all actions in $NAS$,*

*II. for all $P(\vec{s}) \in NAS$ there exists a $P(\vec{w}) \in \mathcal{A}_c(\varphi_c^-(p))$ such that there exists a substitution $\delta$, and $P(\delta(\vec{s}) = P(\delta(\vec{w}))$.*

*III. given any well-formed trace $\widehat{\sigma}'$ such that $\neg a'(i) \supseteq NAS$, then $Tr(\widehat{\sigma}'), i \vDash \forall \vec{x}.\varphi_c^- \supset \varphi_{past}$*

*Proof.*

Given $\widehat{\sigma}_1$, Given any $\vec{t}$ for $\vec{x}$

By Lemma D.13,

there exists a set of inactions $NAS_t$ such that,

$p$ is the performer of all actions in $NAS_t$ (1)

for all $P(\vec{s}) \in NAS_t$ there exists a $P(\vec{w}) \in \mathcal{A}_c(\varphi_c^-(p))$ such that (2)

there exists a substitution $\delta$, and $P(\delta(\vec{s})) = P(\delta(\vec{w}))$ (3)

for all $\widehat{\sigma}, \neg a(i) \supseteq NAS_t, \widehat{\sigma}, i \nVdash \varphi_c^-(p)\{\vec{t}/\vec{x}\}$ (4)

By Lemma D.10,

$Tr(\widehat{\sigma}), i \nvDash \varphi_c^-(p)\{\vec{t}/\vec{x}\}$ (5)

By Definitions of $\vDash$,

$Tr(\widehat{\sigma}), i \vDash (\varphi_c^-(p) \supset \varphi_{past})\{\vec{t}/\vec{x}\}$ (6)

let $NAS = \bigcup_t NAS_t$,

By (1), $I$. holds

By (2), and Lemma D.9 $II$. holds

By (6), Given any $\widehat{\sigma}'$ such that $\neg a(i) \supseteq NAS$

$Tr(\widehat{\sigma}'), k \vDash \forall \vec{x}(\varphi_c^-(p) \supset \varphi_{past})$ (7)

□

**Lemma D.20** (Strong Feasibility of (r1))**.**

$$\begin{aligned}
&\textit{Let } G(a, \neg a, \tau, i, p, t_0) &&= (a = \emptyset) \wedge (\mathrm{dom}(\tau) = \{i\}) \wedge \mathit{start}(\tau) = t_0 \wedge \\
& &&\quad \mathrm{dom}(\neg a) = \{i\} \wedge \forall P \in \neg a(i),\ p \textit{ is the performer of } P \\
&\textit{Let } V(\widehat{\sigma}, i, \Phi) &&= \forall\, \mathbf{G}\ \varphi_i \in \Phi,\ Tr(\widehat{\sigma}), i \vDash \varphi_i
\end{aligned}$$

*For all $j$, $GF'(j, \{\mathbf{G}\ (\forall \vec{x}.\varphi_c^-(p) \supset \varphi_{past})\}, p)$*

*Proof.* By induction on $j$.

Case: $j = 0$

Give any $\widehat{\sigma}_0$ such that,

$\widehat{\sigma}_0 |_p^{\mathcal{A}} = \emptyset, \kappa_0 \supseteq \kappa^p$ (1)

By Lemma D.19,

there exists a set of inactions $NAS$ such that,

$p$ is the performer of all actions in $NAS$, (2)

given any $\widehat{\sigma}'$ such that $\neg a'(i) \supseteq NAS$

if $\widehat{\sigma}'$ is well-formed, then $Tr(\widehat{\sigma}'), i \vDash \forall \vec{x}.\varphi_c^- \supset \varphi_{past}$ (3)

let $a' = \emptyset,, \tau'(0) = \tau_0(0), \neg a'(0) = NAS$

By (2),

$G(a', \neg a', \tau', 0, p, t_0)$ holds (4)

By the actions in $\widehat{\sigma}_0$ and $\neg a'$ belong to different performers,

$\widehat{\sigma}_0 \uplus (\emptyset, \emptyset, a', \neg a', \tau', \emptyset)$ is well-defined (5)

By (3) and (5),

$Tr(\widehat{\sigma}_0 \uplus (\emptyset, \emptyset, a', \neg a', \tau', \emptyset)), 0 \vDash \forall \vec{x}.(\varphi_c^-(p) \supset \varphi_{past})$ (6)

Case: j = k

By I.H. on (k-1),

$$GF'(k-1, \{\forall\vec{x}.\varphi_c^- \supset \varphi_{past}\}, \{p\}) \quad (1)$$

To show $GF'(k, \{\mathbf{G}\ (\forall\vec{x}.\varphi_c^-(p) \supset \varphi_{past})\}, \{p\})$, we unfold $GF'(k-1, \{\mathbf{G}\ (\forall\vec{x}.\varphi_c^-(p) \supset \varphi_{past})\}, \{p\})$, and the first $k-1$ layers of alternating $\forall$ and $\exists$ quantification in $GF'(k, \{\mathbf{G}\ (\forall\vec{x}.\varphi_c^-(p) \supset \varphi_{past})\}, \{p\})$, will be discharged by $GF'(k-1, \{\mathbf{G}\ (\forall\vec{x}.\varphi_c^-(p) \supset \varphi_{past})\}, \{p\})$,

now we are obtained,

$$\forall 0 \le n < k, (\widehat{\sigma}_n\,|_{n-1} = \emptyset)(\widehat{\sigma}_n\,|_n = \widehat{\sigma}_n), \mathit{start}(\tau_n) = \tau_0(0), (\widehat{\sigma}_n\,|_p^{\mathcal{A}} = \emptyset), \kappa_n \supseteq \kappa^p \quad (2)$$

$$\forall 0 \le n < k, a_{np} = \emptyset, \forall P \in, \neg a_{np} \text{ p is the performer of } P,\ \mathrm{dom}(\neg a_{np}) = \mathrm{dom}(\tau_{np}) = \{n\} \quad (3)$$

$$\mathit{start}(\tau_{np}) = \tau_0(0) \quad (4)$$

$$\mathit{Tr}(\widehat{\sigma}_0 \uplus (\emptyset, \emptyset, a_{p0}, \neg a_{p0}, \tau_{p0}, \emptyset) \uplus \cdots \uplus \widehat{\sigma}_n \uplus (\emptyset, \emptyset, a_{pn}, \neg a_{pn}, \tau_{pn}, \emptyset)), n \vDash \forall\vec{x}.\varphi_c^- \supset \varphi_{past}\}, \{p\} \quad (5)$$

Give any $\widehat{\sigma}_k$, such that

$$(\widehat{\sigma}_k\,|_{k-1} = \emptyset)(\widehat{\sigma}_k\,|_k = \widehat{\sigma}_k), (\widehat{\sigma}_k\,|_p^{\mathcal{A}} = \emptyset), \mathit{start}(\tau_k) = \tau_0(0), \kappa_k \supseteq \kappa^p \quad (6)$$

Let $\widehat{\sigma} = \widehat{\sigma}_0 \uplus (\emptyset, \emptyset, a_0', \neg a_0', \tau_0', \emptyset) \uplus \cdots \uplus \widehat{\sigma}_k$,

By Lemma D.19,

there exists a set of inactions $NAS$ such that,

$$p \text{ is the performer of all actions in } NAS, \quad (7)$$

give $\widehat{\sigma}'$ such that $\neg a'(i) \supseteq NAS$

$$\text{if } \widehat{\sigma}' \text{ is well-formed, then } \mathit{Tr}(\widehat{\sigma}'), i \vDash \forall\vec{x}.\varphi_c^- \supset \varphi_{past} \quad (8)$$

let $a' = \emptyset$, $\tau'(k) = \tau_k(k)$, $\neg a'(k) = NAS$

By (7), (2),

$$G(a', \neg a', \tau', k, p, \tau_0(0)) \text{ holds} \quad (9)$$

By the actions in $\widehat{\sigma}_k$ and $\neg a'$ belong to different performers,

$$\widehat{\sigma} \uplus (\emptyset, \emptyset, a', \neg a', \tau', \emptyset) \text{ is well-defined} \quad (10)$$

By (8) and (10),

$$\mathit{Tr}(\widehat{\sigma} \uplus (\emptyset, \emptyset, a', \neg a', \tau', \emptyset)), k \vDash \forall\vec{x}.\varphi_c^- \supset \varphi_{past} \quad (11)$$

□

**Lemma D.21** (Feasibility of (r2) in one state)**.**

*Given state $\widehat{\sigma}$, $i$ such that $\kappa \supseteq \kappa^p$, $\vdash \varphi_{past}$ fin, $\cdot;\cdot \vdash \varphi_f^+(p)$ sat and $\mathcal{A}_f(\varphi_f^+(p)) \vdash \varphi_{past}$, there exists a finite action map $a_p$, and $\tau_p$*

*I. $\forall j \in \mathrm{dom}(a_p)$, $j \ge i$, $\forall j \in \mathrm{dom}(\tau_p)$, $j \ge i$, and $\mathit{start}(\tau_p) = \mathit{start}(\tau)$*

*II. $p$ is the performer of all actions in the range of $a_p$,*

*III. for all $P(\vec{s}) \in range(a_p)$ there exists a $P(\vec{w}) \in \mathcal{A}_f(\varphi_f^+(p))$ such that there exists a substitution $\delta^0$, and $P(\delta^0(\vec{s}) = P(\delta^0(\vec{w}))$.*

*IV. given any $\widehat{\sigma}'$ such that $\widehat{\sigma}'\,|_i = \emptyset$ and $\widehat{\sigma}' \uplus \widehat{\sigma} \uplus (\emptyset, \emptyset, \emptyset, a_p, \emptyset, \tau_p, \emptyset)$ is well-defined, then $\mathit{Tr}(\widehat{\sigma}' \uplus \widehat{\sigma} \uplus (\emptyset, \emptyset, \emptyset, a_p, \emptyset, \tau_p, \emptyset)) \vDash \forall\vec{x}.(\varphi_{past} \supset \varphi_f^+)$*

*Proof.*

Given $\widehat{\sigma}$, $i$, such that $\kappa \supseteq \kappa^p$

By Lemma D.4,

$$\text{there is a finite set of substitutions } \Delta \text{ such that } \forall\delta \in \Delta, \mathit{Tr}(\widehat{\sigma}), i \vDash \delta\varphi_{past} \quad (1)$$

for each $\delta \in \Delta$,

By Lemma D.16,

there exists a finite action map $a_p^\delta$, and $\tau_p^\delta$ such that $\mathit{start}(\tau_p^\delta) = \tau(0)$

$$\forall j \in \mathrm{dom}(a_p^\delta), j \ge i \quad (2)$$

$$p \text{ is the performer of all actions in the range of } a_p \quad (3)$$

for all $P(\vec{s}) \in \mathrm{range}(a_p^\delta)$ there exists a $P(\vec{w}) \in \mathcal{A}_f(\delta\varphi_f^+(p))$ such that

$$\text{there exists a substitution } \delta^0, \text{ and } P(\delta^0(\vec{s})) = P(\delta^0(\vec{w})). \quad (4)$$

$$\text{for all } \widehat{\sigma}_1, \widehat{\sigma}_1 \supseteq (\kappa^p, \emptyset, \emptyset, a_p^\delta, \emptyset, \tau_p^\delta, \emptyset), \widehat{\sigma}, i \Vdash \delta\varphi_f^+(p) \quad (5)$$

let $a_p = \bigcup_\delta a_p^\delta$, $\tau_p = \bigcup_\delta \tau_p^\delta$,
By (2), $I.$ holds
By (3), $II.$ holds
By (4) and Lemma D.9,
$III.$ holds (6)
if $\widehat{\sigma} \uplus (\emptyset, \emptyset, \emptyset, a_p, \emptyset, \tau_p, \emptyset)$ is well-defined,
Given any $\widehat{\sigma}'$, such that $\widehat{\sigma}' |_i = \emptyset$
By (5),
$\widehat{\sigma}' \uplus \widehat{\sigma} \uplus (\emptyset, \emptyset, \emptyset, a_p, \emptyset, \tau_p, \emptyset), i \Vdash \delta\varphi_f^+(p)$ (7)
By Lemma D.10,
$Tr(\widehat{\sigma}' \uplus \widehat{\sigma} \uplus (\emptyset, \emptyset, \emptyset, a_p, \emptyset, \tau_p, \emptyset)), i \vDash \delta\varphi_f^+(p)$ (8)
By Lemma D.6, (6) and Lemma D.2,
$Tr(\widehat{\sigma}' \uplus \widehat{\sigma} \uplus (\emptyset, \emptyset, \emptyset, a_p, \emptyset, \tau_p, \emptyset)), i \vDash \delta\varphi_{past}$ (9)
and $\forall \delta' \notin \Delta$, $Tr(\widehat{\sigma}' \uplus \widehat{\sigma} \uplus (\emptyset, \emptyset, \emptyset, a_p, \emptyset, \tau_p, \emptyset)), i \nvDash \delta'\varphi_{past}$ (10)
By Definitions of $\vDash$, (9) and (8),
$Tr(\widehat{\sigma}' \uplus \widehat{\sigma} \uplus (\emptyset, \emptyset, \emptyset, a_p, \emptyset, \tau_p, \emptyset)), i \vDash \delta(\varphi_{past} \supset \varphi_f^+(p))$ (11)
By Definitions of $\vDash$ and (11),
$Tr(\widehat{\sigma}' \uplus \widehat{\sigma} \uplus (\emptyset, \emptyset, \emptyset, a_p, \emptyset, \tau_p, \emptyset)), i \vDash \forall\vec{x}.(\varphi_{past} \supset \varphi_f^+(p))$ (12)
□

**Lemma D.22** (Strong Feasibility of (r2)).

$$\begin{array}{ll} \textit{Let } G(a, \neg a, \tau, i, p, t_0) & = (\neg a = \emptyset) \wedge \forall k \in \mathrm{dom}(a), k \geq i \wedge \forall k \in \mathrm{dom}(\tau), k \geq i \wedge \textit{start}(\tau) = t_0 \wedge \\ & \quad \forall k \in \mathrm{dom}(a), \forall P \in a(k),\ p \textit{ is the performer of } P \\ \textit{Let } V(\widehat{\sigma}, i, \Phi) & = \forall\, \mathbf{G}\ \varphi_i \in \Phi,\ Tr(\widehat{\sigma}), i \vDash \varphi_i \end{array}$$

*If* $\vdash \varphi_{past}$ fin, $\cdot;\cdot \vdash \varphi_f^+$ *and* $\mathcal{A}_f(\varphi_f^+(p)) \vdash \varphi_{past}$, *then for all* $j$, $GF'(j, \{\mathbf{G}\ (\forall\vec{x}.\varphi_{past} \supset \varphi_f^+(p))\}, p)$

*Proof.* By induction on $j$.

Case: $j = 0$
Give any $\widehat{\sigma}_0$ such that,
$\widehat{\sigma}_0 |_p^{\mathcal{A}} = \emptyset, \kappa_0 \supseteq \kappa^p$ (1)
By Lemma D.21,
there exists a set of inactions a finite action map $a'$, and $\tau'$ such that,
$\textit{start}(\tau') = \tau_0(0)$ (2)
$\forall n \in \mathrm{dom}(a'), n \geq i, p$ is the performer of all actions in the range of $a'$, (3)
given any $\widehat{\sigma}''$ such that $\widehat{\sigma}'' |_0 = \emptyset$ if $\widehat{\sigma}' \uplus \widehat{\sigma}_0 \uplus (, \emptyset, \emptyset, a', \emptyset, \tau', \emptyset)$ is well-defined
then $\widehat{\sigma}'' \uplus \widehat{\sigma}_0 \uplus (\emptyset, \emptyset, \emptyset, a', \emptyset, \tau', \emptyset), 0 \vDash \forall\vec{x}.\varphi_{past} \supset \varphi_f^+$ (4)
let $\neg a' = \emptyset$,
By (3), (2),
$G(a', \neg a', \tau', 0, p, \tau_0(0))$ holds (5)
By the actions in $\widehat{\sigma}_0$ and $\neg a'$ belong to different performers,
$\widehat{\sigma}_0 \uplus (\emptyset, \emptyset, a', \neg a', \tau', \emptyset)$ is well-defined (6)
By (4),
$Tr(\widehat{\sigma}'' \uplus \widehat{\sigma}_0 \uplus (\emptyset, \emptyset, a', \neg a', \tau', \emptyset)), 0 \vDash \forall\vec{x}.\varphi_{past} \supset \varphi_f^+(p)$ (7)

Case: j = k
By I.H. on (k-1),
$GF'(k-1, \{\forall\ \mathbf{G}\ (\vec{x}.\varphi_{past} \supset \varphi_f^+(p))\}, \{p\})$ (1)
To show $GF'(k, \{\mathbf{G}\ (\forall\vec{x}.\varphi_{past} \supset \varphi_f^+(p))\}, \{p\})$, we unfold $GF'(k-1, \{\mathbf{G}\ (\forall\vec{x}.\varphi_{past} \supset \varphi_f^+(p))\}, \{p\})$,
and the first $k-1$ layers of alternating $\forall$ and $\exists$ quantification in $GF'(k, \{\mathbf{G}\ (\forall\vec{x}.\varphi_{past} \supset \varphi_f^+(p))\}, \{p\})$,
will be discharged by $GF'(k-1, \{\mathbf{G}\ (\forall\vec{x}.\varphi_{past} \supset \varphi_f^+(p))\}, \{p\})$,

now we are obtained,

$$\forall 0 \le n < k, (\widehat{\sigma}_n\,|_{n-1}= \emptyset)(\widehat{\sigma}_n\,|_n= \widehat{\sigma}_n), \mathit{start}(\tau_n) = \tau_0(0), (\widehat{\sigma}_n\,|_p^{\mathcal{A}}= \emptyset), \kappa_n \supseteq \kappa^p \tag{2}$$

$$(\neg a_{np} = \emptyset), \forall m \in \mathrm{dom}(a_{np}), m \ge n \tag{3}$$

$$\mathit{start}(\tau_{np}) = \tau_0(0) \tag{4}$$

$$\forall P \in \mathrm{range}(a_{pn}), p \text{ is the performer of } P \tag{5}$$

Give any $\widehat{\sigma}_k$, such that

$$(\widehat{\sigma}_k\,|_{k-1}= \emptyset)(\widehat{\sigma}_k\,|_k= \widehat{\sigma}_k), (\widehat{\sigma}_k\,|_p^{\mathcal{A}}= \emptyset), \mathit{start}(\tau_k) = \tau_0(0), \kappa_k \supseteq \kappa^p \tag{6}$$

Let $\widehat{\sigma} = \widehat{\sigma}_0 \uplus (\emptyset, \emptyset, a'_0, \neg a'_0, \tau'_0, \emptyset) \uplus \cdots \uplus \widehat{\sigma}_k$,

By Lemma D.21,

there exists a set of inactions a finite action map $a'$, and $\tau'$ such that,

$$\forall n \in \mathrm{dom}(a'), n \ge i, p \text{ is the performer of all actions in the range of } a', \tag{7}$$

$$\forall n \in \tau(a'), n \ge i, \text{ and } \mathit{start}(\tau') = \tau_0(0) \tag{8}$$

given $\widehat{\sigma}''$ such that $\widehat{\sigma}''\,|_k= \emptyset$ if $\widehat{\sigma}'' \uplus \widehat{\sigma} \uplus (, \emptyset, \emptyset, a', \emptyset, \tau', \emptyset)$ is well-defined

$$\text{then } \widehat{\sigma}'' \uplus \widehat{\sigma} \uplus (\emptyset, \emptyset, \emptyset, a', \emptyset, \tau', \emptyset), k \vDash \forall \vec{x}.\varphi_{past} \supset \varphi_f^+ \tag{9}$$

let $\neg a' = \emptyset$,

By (7), (8),

$$G(a', \neg a', \tau', 0, p, \tau_0(0)) \text{ holds} \tag{10}$$

By the actions in $\widehat{\sigma}_k$ and $\neg a'$ belong to different performers,

$$\widehat{\sigma} \uplus (\emptyset, \emptyset, a', \neg a', \tau', \emptyset) \text{ is well-defined} \tag{11}$$

By (9),

$$\mathit{Tr}(\widehat{\sigma}'' \uplus \widehat{\sigma} \uplus (\emptyset, \emptyset, a', \neg a', \tau', \emptyset)), 0 \vDash \forall \vec{x}.\varphi_{past} \supset \varphi_f^+(p) \tag{12}$$

□

**Lemma D.23** (Feasibility of (r3) in one state).

*Given state $i$, $\widehat{\sigma}$ such that $\kappa \supseteq \kappa^p$,*

$\cdot;\cdot \vdash \varphi_f^-(p)$ sat *there exists a finite action map $\neg a_p$, and $\tau_p$*

I. *$\forall j \in \mathrm{dom}(\neg a_p)$, $j \ge i$, $\forall j \in \mathrm{dom}(\tau_p)$, $j \ge i$, and $\mathit{start}(\tau_p) = \mathit{start}(\tau)$*

II. *$p$ is the performer of all actions in the range of $\neg a_p$,*

III. *for all $P(\vec{s}) \in \mathit{range}(\neg a_p)$ there exists a $P(\vec{w}) \in \mathcal{A}_f(\varphi_f^-(p))$ such that there exists a substitution $\delta^0$, and $P(\delta^0(\vec{s}) = P(\delta^0(\vec{w}))$.*

IV. *given any $\widehat{\sigma}'$ such that $\widehat{\sigma}'\,|_i= \emptyset$ and $\widehat{\sigma}' \uplus \widehat{\sigma} \uplus (\emptyset, \emptyset, \emptyset, a_p, \emptyset, \tau_p, \emptyset)$ is well-defined, then $\widehat{\sigma}' \uplus \widehat{\sigma} \uplus (\emptyset, \emptyset, \emptyset, \emptyset, \neg a_p, \tau_p, \emptyset) \vDash \forall \vec{x}.\varphi_{past} \supset \varphi_f^-$*

**Proof** (sketch): Similar to the proof of Lemma D.19 □

**Lemma D.24** (Strong Feasibility of (r3)).

$$\begin{aligned} &\textit{Let } G(a, \neg a, \tau, i, p, t_0) &&= (\neg a = \emptyset) \wedge \forall k \in \mathrm{dom}(a), k \ge i \wedge \forall k \in \mathrm{dom}(\tau), k \ge i \wedge \mathit{start}(\tau) = t_0 \wedge \\ & &&\quad \forall k \in \mathrm{dom}(a), \forall P \in a(k),\ p \textit{ is the performer of } P \\ &\textit{Let } V(\widehat{\sigma}, i, \Phi) &&= \forall\, \mathbf{G}\, \varphi_i \in \Phi,\ \mathit{Tr}(\widehat{\sigma}), i \vDash \varphi_i \end{aligned}$$

*For all $j$, $GF'(j, \{\mathbf{G}\ (\forall \vec{x}.\varphi_{past} \supset \varphi_f^-(p))\}, p)$*

**Proof** (sketch): Similar to the proof of Lemma D.20 □

**Theorem D.25** (Feasibility of a single responsibility).

*F1.* $\mathbf{G}\ \forall \vec{x}.\varphi_c^-(p) \supset \varphi_{past}$ *is feasible for agent $p$*

*F2.* $\mathbf{G}\ \forall \vec{x}.\varphi_{past} \supset \varphi_f^+(p)$ *is feasible for agent $p$ if* $\vdash \varphi_{past}$ fin, $\cdot;\cdot \vdash \varphi_f^+$ *and* $\mathcal{A}_f(\varphi_f^+(p)) \vdash \varphi_{past}$

*F3.* $\mathbf{G}\ \forall \vec{x}.\varphi_{past} \supset \varphi_f^-(p)$ *is feasible*

*Proof.*
*F1.* By Lemma D.20.
*F2.* By Lemma D.22.
*F3.* By Lemma D.24. $\square$

**Lemma D.26** (Feasibility compositions for one agent in one state)**.** *Let $\Phi$ be a set of responsibilities, and $\Phi = \Phi_c, \Phi_{f+}, \Phi_{f-}$ where $\forall\ \mathbf{G}\ \varphi \in \Phi_c$, $\varphi$ is of the form $\forall\vec{x}.\varphi_c^-(p) \supset \varphi_{past}$, $\forall\ \mathbf{G}\ \varphi \in \Phi_{f+}$, $\varphi$ is of the form $\forall\vec{x}.\varphi_{past} \supset \varphi_f^+(p)$, and $\forall\ \mathbf{G}\ \varphi \in \Phi_{f-}$, $\varphi$ is of the form $\forall\vec{x}.\varphi_{past} \supset \varphi_f^-(p)$.*

1. *for all $\mathbf{G}\ (\forall\vec{x}.\varphi_{past} \supset \varphi_f^-(p)) \in \Phi_{f-}$, $\cdot;\cdot \vdash \varphi_f^-\ \mathsf{sat}$*
2. *for all $\mathbf{G}\ (\forall\vec{x}.\varphi_{past} \supset \varphi_f^+(p)) \in \Phi_{f+}$, $\vdash \varphi_{past}\ \mathsf{fin}$, $\cdot;\cdot \vdash \varphi_f^+\ \mathsf{sat}$ and $\mathcal{A}_f(\varphi_f^+(p)) \vdash \varphi_{past}$*
3. *for any two responsibilities $\mathbf{G}\ (\forall\vec{x_1}.\varphi_{past1} \supset \varphi_{f1}^+(p)) \in \Phi_{f+}$, $\mathbf{G}\ (\forall\vec{x_2}.\varphi_{past2} \supset \varphi_{f2}^+(p)) \in \Phi_{f+}$, $\mathcal{A}_f(\varphi_{fj}^+(p)) \vdash \varphi_{pasti}$ $(i = 1, 2,\ j = 1, 2,\ i \neq j)$*
4. *for any two responsibilities $\mathbf{G}\ (\forall\vec{x_1}.\varphi_{past1} \supset \varphi_{f1}^+(p)) \in \Phi_{f+}$, $\mathbf{G}\ (\forall\vec{x_2}.\varphi_{past2} \supset \varphi_{f2}^-(p)) \in \Phi_{f-}$, for all $\delta$, $\delta\mathcal{A}_f(\varphi_{f1}^+(p)) \cap \delta\mathcal{A}_f(\varphi_{f2}^+(p)) = \emptyset$*
5. *for any two responsibilities $\mathbf{G}\ (\forall\vec{x_1}.\varphi_{past1} \supset \varphi_{f1}^+(p)) \in \Phi_{f+}$, $\mathbf{G}\ (\forall\vec{x_2}.\varphi_{c2}^-(p) \supset \varphi_{past2}) \in \Phi_c$,*
   - *(a) either for all $\delta$, $\delta\mathcal{A}_f(\varphi_{f1}^+(p)) \cap \delta\mathcal{A}_c(\varphi_{c2}^-(p)) = \emptyset$*
   - *(b) or $\varphi_{past1} = \varphi_P$, and for all $P \in \mathcal{A}_c(\varphi_{c2}^-(p))$, for each mgu $\delta$ such that $\delta P \in \delta(\mathcal{A}_f(\varphi_{f1}^+(p)))$, $\delta\varphi_p \vdash \delta\varphi_{past2}$*

*Given any $\widehat{\sigma}$, $i$ such that*

1. *$\kappa \supseteq \kappa^p$,*
2. *$\forall j \geq i$, $\forall P \in a(j)$ such that $p$ is the performer of $P$, $\exists\ \mathbf{G}\ \varphi_i \in \Phi_{f+}$ and $\varphi_i = \forall\vec{x}.\varphi_{past} \supset \varphi_f^+(p)$, $\exists k$, $\exists\delta$ such that $k < i$, and $\widehat{\sigma}, k \vDash \delta\varphi_{past}$, and $\widehat{\sigma}, k \Vdash \delta\varphi_f^+$ $\exists P' \in \mathcal{A}_f(\delta\varphi_f^+(p))$, and $\exists\delta^0$, and $P = \delta^0 P'$*
3. *$\forall j \geq i$, $\forall P \in \neg a(j)$ such that $p$ is the performer of $P$, $\exists\ \mathbf{G}\ \varphi_i \in \Phi_{f-}$ and $\varphi_i = \forall\vec{x}.\varphi_{past} \supset \varphi_f^-(p)$, $\exists P' \in \mathcal{A}_f(\varphi_f^-(p))$, $\exists\delta^0$, and $P = \delta^0 P'$*

*there exists $a_p$, $\neg a_p$, $\tau_p$, such that*

1. *$\forall k \in \mathrm{dom}(a_p), k \geq i$, $\forall k \in \mathrm{dom}(\neg a_p), k \geq i \wedge$ and $\forall k \in \mathrm{dom}(\tau_p), k \geq i$ and $start(\tau_p) = start(\tau)$*
2. *$\forall P \in range(a_p)$, $p$ is the performer of $P$ $\wedge$*
3. *$\forall P \in range(\neg a_p)$, $p$ is the performer of $P$ $\wedge$*
4. *$\forall k \in \mathrm{dom}(a_p), \forall P \in a_p(k)$, $\exists\ \mathbf{G}\ (\forall\vec{x}.\varphi_{past} \supset \varphi_f^+(p)) \in \Phi_{f+}$, $\exists\delta$ such that $\widehat{\sigma} \uplus (\emptyset, \emptyset, a_p, \neg a_p, \tau_p, \emptyset), i \vDash \delta\varphi_{past}$, and $\widehat{\sigma} \uplus (\emptyset, \emptyset, a_p, \neg a_p, \tau_p, \emptyset), i \Vdash \delta\varphi_f^+$ $\exists P' \in \mathcal{A}_f(\delta\varphi_f^+(p))$, and $\exists\delta^0$, and $P = \delta^0 P'$*
5. *$\forall P \in (\neg a_p(i))$, $\exists\ \mathbf{G}\ \varphi_i \in \Phi_{f-}$, such that $\exists P' \in \mathcal{A}_f(\varphi_f^-(p)) \cup \mathcal{A}_c(\varphi_c^-(p))$, $\exists\delta^0$, and $\delta^0 P = \delta^0 P'$*
6. *$\forall k \in \mathrm{dom}(\neg a_p), k > i, \forall P \in a_p(k)$, $\exists P' \in \mathcal{A}_f(\varphi_f^-(p))$, $\exists\delta^0$, and $\delta^0 P = \delta^0 P'$.*
7. *$\widehat{\sigma} \uplus (\emptyset, \emptyset, a_p, \neg a_p, \tau_p, \emptyset))$ is well-defined,*
8. *given any $\widehat{\sigma}'$ such that $\widehat{\sigma}' |_i = \emptyset$, $\forall\varphi_i \in \Phi$, if $\widehat{\sigma}' \uplus \widehat{\sigma} \uplus (\emptyset, \emptyset, a_p, \neg a_p, \tau_p, \emptyset)$ is well-defined then $\widehat{\sigma}' \uplus \widehat{\sigma} \uplus (\emptyset, \emptyset, a_p, \neg a_p, \tau_p, \emptyset), i \Vdash \varphi_i$*

*Proof.*
Given any $\widehat{\sigma}$, $i$, such that

$$\kappa \supseteq \kappa^p \tag{1}$$

$$\begin{aligned}&\forall j \geq i, \forall P \in a(j) \text{ such that } p \text{ is the performer of } P,\\ &\exists\ \mathbf{G}\ \varphi_i \in \Phi_{f+} \text{ and } \varphi_i = \forall\vec{x}.\varphi_{past} \supset \varphi_f^+(p), \exists k, \exists\delta \text{ such that } k < i,\\ &\text{ and } \widehat{\sigma}, k \vDash \delta\varphi_{past}, \widehat{\sigma}, k \Vdash \delta\varphi_f^+, \text{ and } \exists P' \in \mathcal{A}_f(\delta\varphi_f^+(p)), \text{ and } \exists\delta^0, such that P = \delta^0 P'\end{aligned} \tag{2}$$

$\forall j \geq i, \forall P \in \neg a(j)$ such that $p$ is the performer of $P$,

$$\exists\, \mathbf{G}\, \varphi_i \in \Phi_{f-} \text{ and } \varphi_i = \forall \vec{x}.\varphi_{past} \supset \varphi_f^-(p), \exists P' \in \mathcal{A}_f(\varphi_f^-(p)), \exists \delta^0, \text{ and } \delta^0 P = \delta^0 P' \quad (3)$$

let $t^0 = \mathit{start}(\tau)$,
Given any $\varphi_{r1} = \forall \vec{x}.\varphi_c^-(p) \supset \varphi_{past}$ such that $\mathbf{G}\, \varphi_{r1} \in \Phi_c$,
Given any $\vec{t}$ for $\vec{x}$,
By Lemma D.13,
there exists a set of inactions $NAS_t$ such that,

$$p \text{ is the performer of all actions in } NAS_t \quad (4)$$

for all $P(\vec{s}) \in NAS_t$ there exists a $P(\vec{w}) \in \mathcal{A}_c(\varphi_c^-(p))$ such that

$$\text{there exists a substitution } \delta, \text{ and } P(\delta(\vec{s})) = P(\delta(\vec{w})) \quad (5)$$

$$\text{for all } \widehat{\sigma}, \neg a(i) \supseteq NAS_t, \widehat{\sigma}, i \nVdash \varphi_c^-(p)\{\vec{t}/\vec{x}\} \quad (6)$$

Given any $\varphi_{r2} = \forall \vec{x}.\varphi_{past} \supset \varphi_f^+(p)$ such that $\mathbf{G}\, \varphi_{r2} \in \Phi_{f+}$,
By Lemma D.4,

$$\text{there is a finite set of substitutions } \Delta \text{ such that } \forall \delta \in \Delta, \mathit{Tr}(\widehat{\sigma}), i \vDash \delta\varphi_{past} \quad (7)$$

for each $\delta \in \Delta$,
By Lemma D.16,
there exists a finite action map $a_p^\delta$, and $\tau_p^\delta \succ t^0$

$$\forall j \in \mathrm{dom}(a_p^\delta), j \geq i, \forall j \in \mathrm{dom}(\tau_p^\delta), j \geq i \quad (8)$$

$$p \text{ is the performer of all actions in the range of } a_p \quad (9)$$

for all $P(\vec{s}) \in \mathrm{range}(a_p)$ there exists a $P(\vec{w}) \in \mathcal{A}_f(\delta\varphi_f^+(p))$ such that

$$\text{there exists a substitution } \delta^0, \text{ and } P(\delta^0(\vec{s})) = P(\delta^0(\vec{w})). \quad (10)$$

$$\text{for all well-formed } \widehat{\sigma}_1, \widehat{\sigma}_1 \supseteq (\kappa^p, \emptyset, \emptyset, a_p^\delta, \emptyset, \tau_p^\delta, \emptyset), \widehat{\sigma}_1, i \Vdash \delta\varphi_f^+(p) \quad (11)$$

let $a'_p = \bigcup_\delta a_p^\delta$, $\tau_{p1} = \bigsqcup_\delta \tau_p^\delta$,
Given any $\varphi_{r3} = \forall \vec{x}.\varphi_{past} \supset \varphi_f^-(p)$ such that $\mathbf{G}\, \varphi_{r2} \in \Phi_{f+}$,
for each $\vec{t}$ for $\vec{x}$,
By Lemma D.18,
there exists a map $\neg a_t$, and $\tau_t \succ t^0$

$$\forall j \in \mathrm{dom}(a_t), j \geq i, \forall j \in \mathrm{dom}(\tau_t), j \geq i \quad (12)$$

$$p \text{ is the performer of all actions in the range of } a_t \quad (13)$$

for all $P(\vec{s}) \in \mathrm{range}(a_t)$ there exists a $P(\vec{w}) \in \mathcal{A}_f(\delta\varphi_f^-(p))$ such that

$$\text{there exists a substitution } \delta^0, \text{ and } P(\delta^0(\vec{s})) = P(\delta^0(\vec{w})). \quad (14)$$

$$\text{for all well-formed } \widehat{\sigma}_1, \widehat{\sigma}_1 \supseteq (\kappa^p, \emptyset, \emptyset, \emptyset, a_t, \emptyset, \tau_t, \emptyset), \widehat{\sigma}_1, i \Vdash \delta\varphi_f^-(p) \quad (15)$$

let $\neg a'_p = \bigcup_t a_t$, $\tau_{p2} = \bigsqcup_t \tau_t$,
let $a_p = a'_p$, $\neg a_p = \neg a'_p \cup \{i \mapsto \bigcup_t NAS_t \backslash \{a'_p(i) \cup a(i)\}\}$, $\tau_p = \tau_{p1} \sqcup \tau_{p2}$
By (8), (12), 1 holds
By (4), (9), (13), 2 and 3 hold
By (10), 4 holds
By (5), (14), 5 and 6 hold
By assumption 3 about $\varphi_i$, (5), (10) and (14),

$$\forall k, a_p(k) \cap \neg a_p(k) = \emptyset \quad (16)$$

By the definition of $\neg a_p$,

$$\forall k, a(k) \cap \neg a_p(k) = \emptyset \quad (17)$$

By assumption 3 about $\varphi_i$, (3) and (10),

$$\forall k, \neg a(k) \cap a_p(k) = \emptyset \quad (18)$$

By (16), (17) and (18),

$$\widehat{\sigma} \uplus (\emptyset, \emptyset, a_p, \neg a_p, \tau_p, \emptyset) \text{ is well-defined} \quad (19)$$

Given any $\widehat{\sigma}'$, such that $\widehat{\sigma}'|_i = \emptyset$, and given any $\mathbf{G}\, \forall \vec{x}.\varphi_{past} \supset \varphi_f^-(p) \in \Phi_{f-}$
By (15),

$$\widehat{\sigma}' \uplus \widehat{\sigma} \uplus (\emptyset, \emptyset, a_p, \neg a_p, \tau_p, \emptyset) \Vdash \forall \vec{x}.\varphi_{past} \supset \varphi_f^-(p) \quad (20)$$

Given any $\widehat{\sigma}'$, such that $\widehat{\sigma}'|_i = \emptyset$, and given any $\mathbf{G}\, \forall \vec{x}.\varphi_{past} \supset \varphi_f^+(p) \in \Phi_{f+}$
By Lemma D.6, (7) and assumption 3 about $\varphi_i$, and Lemma D.2

given the finite set $\Delta$ such that $\forall\delta \in \Delta$, $Tr(\widehat{\sigma}), i \vDash \delta\varphi_{past}$ and $\forall\delta' \notin \Delta$, $Tr(\widehat{\sigma}), i \nvDash \delta'\varphi_{past}$

$$Tr(\widehat{\sigma}' \uplus \widehat{\sigma} \uplus (\emptyset, \emptyset, \emptyset, a_p, \neg a_p, \tau_p, \emptyset)), i \vDash \delta\varphi_{past} \tag{21}$$

$$\text{and } \forall\delta' \notin \Delta,\ Tr(\widehat{\sigma}' \uplus \widehat{\sigma} \uplus (\emptyset, \emptyset, \emptyset, a_p, \neg a_p, \tau_p, \emptyset)), i \nvDash \delta'\varphi_{past} \tag{22}$$

By (10),

$$\widehat{\sigma}' \uplus \widehat{\sigma} \uplus (\emptyset, \emptyset, a_p, \neg a_p, \tau_p, \emptyset) \Vdash \varphi_f^+(p) \tag{23}$$

By Definitions of $\Vdash$,(21), (22) and (23),

$$Tr(\widehat{\sigma}' \uplus \widehat{\sigma} \uplus (\emptyset, \emptyset, \emptyset, a_p, \neg a_p, \tau_p, \emptyset)), i \Vdash \forall\vec{x}.(\varphi_{past} \supset \varphi_f^+(p)) \tag{24}$$

Given any $\widehat{\sigma}'$, such that $\widehat{\sigma}'|_i = \emptyset$, and given any $\mathbf{G}\ \forall\vec{x}_1.\varphi_c^-(p) \supset \varphi_{past1} \in \Phi_c$

Given any $\delta_{x1}$ for $\vec{x}_1$, there are three cases $(i)$, $(ii)$ and $(iii)$

$(i)$. $NAS_t \cap (a_p'(i) \cup a(i)) = \emptyset$,

By (6),

$$\widehat{\sigma}' \uplus \widehat{\sigma} \uplus (\emptyset, \emptyset, a_p, \neg a_p, \tau_p, \emptyset) \nVdash \delta_{x1}\varphi_c^-(p) \tag{25}$$

$(ii)$. $\exists P \in NAS_t \cap (a_p(i))$,

By (5),

$$\exists P_1 \in \mathcal{A}_c(\delta_{x1}\varphi_c^-(p)) \text{ such that } \exists\delta_1 \text{ and } P = \delta_1 P_1 \tag{26}$$

By Lemma D.9,

$$\exists P_1' \in \mathcal{A}_c(\varphi_c^-(p)) \text{ such that } P_1 = \delta_{x1} P_1' \tag{27}$$

By the definition of $a_p$ and (10),

$$\exists\ \mathbf{G}\ (\forall\vec{x}_2.\varphi_{past2} \supset \varphi_f^+(p)) \in \Phi_{f+} \text{ such that } \exists\delta_{x2} \text{ and} Tr(\widehat{\sigma}), i \vDash \delta_{x2}(\varphi_{past2}), \text{ and} \tag{28}$$

$$\exists P_2 \in \mathcal{A}_f(\delta_{x2}(\varphi_f^+(p))) \text{ such that } \exists\delta_2 \text{ and } P = \delta_2 P_2 \tag{29}$$

By Lemma D.9,

$$\exists P_2' \in \mathcal{A}_f(\varphi_f^+(p)) \text{ such that } P_2 = \delta_{x2} P_2' \tag{30}$$

By (26), (27), (29) and (30),

there exists a most general unifier $\delta^0$ for $P_1'$ and $P_2'$, such that

$$\delta_1\delta_{x1} = \delta'\delta^0, \delta_2\delta_{x2} = \delta'\delta^0, \tag{31}$$

By assumption $4(b)$ about $\varphi_i \in \Phi$,

$$\varphi_{past2} = \varphi_P \text{ and } \delta^0\varphi_{past2} \vdash \delta^0\varphi_{past1} \tag{32}$$

By substitution lemma on the proof rules,

$$\delta'\delta^0\varphi_{past2} \vdash \delta'\delta^0\varphi_{past1} \tag{33}$$

By Lemma D.6, (28) and assumption 2 about $\varphi_i$ and Lemma D.2

$$Tr(\widehat{\sigma}' \uplus \widehat{\sigma} \uplus (\emptyset, \emptyset, \emptyset, a_p, \neg a_p, \tau_p, \emptyset)), i \vDash \delta_{x2}\varphi_{past2} \tag{34}$$

By Proof theory is sound, (34), (31) and $\delta_{x1}\varphi_{past1}$ is closed,

$$Tr(\widehat{\sigma}' \uplus \widehat{\sigma} \uplus (\emptyset, \emptyset, \emptyset, a_p, \neg a_p, \tau_p, \emptyset)), i \vDash \delta_{x1}\varphi_{past1} \tag{35}$$

$(iii)$. $\exists P \in NAS_t \cap a(i))$,

By (5),

$$\exists P_1 \in \mathcal{A}_c(\delta_{x1}\varphi_c^-(p)) \text{ such that } \exists\delta_1 \text{ and } P = \delta_1 P_1 \tag{36}$$

By Lemma D.9,

$$\exists P_1' \in \mathcal{A}_c(\varphi_c^-(p)) \text{ such that } P_1 = \delta_{x1} P_1' \tag{37}$$

By (2),

$$\exists\ \mathbf{G}\ (\forall\vec{x}_2.\varphi_{past2} \supset \varphi_f^+(p)) \in \Phi_{f+} \text{ such that } \exists k, k < i, \exists\delta_{x2} \text{ and } Tr(\widehat{\sigma}), i \vDash \delta_{x2}(\varphi_{past2}), \text{ and} \tag{38}$$

$$\widehat{\sigma}, k \Vdash \delta_{x2}\varphi_{f2}^+ \text{ and } \exists P_2 \in \mathcal{A}_f(\delta_{x2}(\varphi_f^+(p))) \text{ such that } \exists\delta_2 \text{ and } P = \delta_2 P_2 \tag{39}$$

By Lemma D.9,

$$\exists P_2' \in \mathcal{A}_f(\varphi_f^+(p)) \text{ such that } P_2 = \delta_{x2} P_2' \tag{40}$$

By (37), (38), (39) and (40),

there exists a most general unifier $\delta^0$ for $P_1'$ and $P_2'$, such that

$$\delta_1\delta_{x1} = \delta'\delta^0, \delta_2\delta_{x2} = \delta'\delta^0, \tag{41}$$

By assumption $5(b)$ about $\varphi_i \in \Phi$,

$$\varphi_{past2} = \varphi_P \text{ and } \delta^0\varphi_{past2} \vdash \delta^0\varphi_{past1} \tag{42}$$

By substitution lemma on the proof rules,

$$\delta'\delta^0\varphi_{past2} \vdash \delta'\delta^0\varphi_{past1} \tag{43}$$

By Lemma D.6, (39) and assumption 3 about $\varphi_i$ and Lemma D.2

$$Tr(\widehat{\sigma}' \uplus \widehat{\sigma} \uplus (\emptyset, \emptyset, \emptyset, a_p, \neg a_p, \tau_p, \emptyset)), k \vDash \delta_{x2}\varphi_{past2} \tag{44}$$

By $\varphi_{past2} = \varphi_P$ and Lemma D.3,

$$Tr(\widehat{\sigma}' \uplus \widehat{\sigma} \uplus (\emptyset, \emptyset, \emptyset, a_p, \neg a_p, \tau_p, \emptyset)), i \vDash \delta_{x2}\varphi_{past2} \tag{45}$$

By Proof theory is sound, (45), (43) and $\delta_{x1}\varphi_{past1}$ is closed,

$$Tr(\widehat{\sigma}' \uplus \widehat{\sigma} \uplus (\emptyset, \emptyset, \emptyset, a_p, \neg a_p, \tau_p, \emptyset)), i \vDash \delta_{x1}\varphi_{past1} \tag{46}$$

By (25), (35), (46),

$$Tr(\widehat{\sigma}' \uplus \widehat{\sigma} \uplus (\emptyset, \emptyset, \emptyset, a_p, \neg a_p, \tau_p, \emptyset)), i \Vdash \forall \vec{x}_1.\varphi^-_{c1}(p) \supset \varphi_{past1} \tag{47}$$

□

We make a small change to the structure of $GF'$. Now the abstract predicate $G$ takes an additionally argument $\widehat{\sigma}$. In the definition of $GF'$, $G$ is supplied with $\biguplus_{k=0}^{j} \widehat{\sigma}_k \uplus \biguplus_{k=0}^{j-1}(\emptyset, \emptyset, a'_k, \neg a'_k, \tau'_k, \emptyset)$ at level $j$.

**Lemma D.27** (Strong feasibility compositions for one agent)**.** *Let $\Phi$ be a set of responsibilities, and $\Phi = \Phi_c, \Phi_{f+}, \Phi_{f-}$ where $\forall\ \mathbf{G}\ \varphi \in \Phi_c$, $\varphi$ is of the form $\forall \vec{x}.\varphi^-_c(p) \supset \varphi_{past}$, $\forall\ \mathbf{G}\ \varphi \in \Phi_{f+}$, $\varphi$ is of the form $\forall \vec{x}.\varphi_{past} \supset \varphi^+_f(p)$, and $\forall\ \mathbf{G}\ \varphi \in \Phi_{f-}$, $\varphi$ is of the form $\forall \vec{x}.\varphi_{past} \supset \varphi^-_f(p)$.*

$$\begin{array}{lll}
\textit{Let } G(\widehat{\sigma}, a, \neg a, \tau, i, p, t_0) & = & \forall k \in \mathrm{dom}(a), k \geq i \wedge \forall k \in \mathrm{dom}(\neg a), k \geq i \wedge \forall k \in \mathrm{dom}(\tau), k \geq i \wedge \mathit{start}(\tau) = t_0 \wedge \\
& & \forall P \in \mathit{range}(a),\ p \textit{ is the performer of } P\ \wedge \\
& & \forall P \in \mathit{range}(\neg a),\ p \textit{ is the performer of } P\ \wedge \\
& & \forall k \in \mathrm{dom}(a), \forall P \in a(k), \\
& & \quad \exists\ \mathbf{G}\ \varphi_i \in \Phi_{f+}, \exists \delta \textit{ such that } \widehat{\sigma}, i \vDash \delta\varphi_{past}, \textit{ and} \\
& & \quad \widehat{\sigma}, i \Vdash \delta\varphi^+_f\ \exists P' \in \mathcal{A}_f(\delta\varphi^+_f(p)), \textit{ and } \exists \delta^0, \textit{ and } P = \delta^0 P' \\
& & \forall P \in (\neg a(i)), \exists P' \in \mathcal{A}_f(\varphi^-_f(p)) \cup \mathcal{A}_c(\varphi^-_c(p)), \exists \delta^0, \textit{ and } \delta^0 P = \delta^0 P' \\
& & \forall k \in \mathrm{dom}(\neg a), k > i, \forall P \in a(k), \\
& & \quad \exists P' \in \mathcal{A}_f(\varphi^-_f(p)), \exists \delta^0, \textit{ and } \delta^0 P = \delta^0 P'. \\
\textit{Let } V(\widehat{\sigma}, i, \Phi) & = & \forall\ \mathbf{G}\ \varphi_i \in \Phi,\ Tr(\widehat{\sigma}), i \Vdash \varphi_i
\end{array}$$

*then for all $j$, $GF'(j, \Phi, p)$ if*

1. *for all $\mathbf{G}\ (\forall \vec{x}.\varphi_{past} \supset \varphi^-_f(p)) \in \Phi_{f-}$, $\cdot;\cdot \vdash \varphi^-_f\ \mathsf{sat}$*
2. *for all $\mathbf{G}\ (\forall \vec{x}.\varphi_{past} \supset \varphi^+_f(p)) \in \Phi_{f+}$, $\vdash \varphi_{past}\ \mathsf{fin}$, $\cdot;\cdot \vdash \varphi^+_f$ and $\mathcal{A}_f(\varphi^+_f(p)) \vdash \varphi_{past}$*
3. *for any two responsibilities $\mathbf{G}\ (\forall \vec{x_1}.\varphi_{past1} \supset \varphi^+_{f1}(p)) \in \Phi_{f+}$, $\mathbf{G}\ (\forall \vec{x_2}.\varphi_{past2} \supset \varphi^+_{f2}(p)) \in \Phi_{f+}$, $\mathcal{A}_f(\varphi^+_{fj}(p)) \vdash \varphi_{pasti}$ $(i = 1, 2,\ j = 1, 2,\ i \neq j)$*
4. *for any two responsibilities $\mathbf{G}\ (\forall \vec{x_1}.\varphi_{past1} \supset \varphi^+_{f1}(p)) \in \Phi_{f+}$, $\mathbf{G}\ (\forall \vec{x_2}.\varphi_{past2} \supset \varphi^-_{f2}(p)) \in \Phi_{f-}$, for all $\delta$, $\delta\mathcal{A}_f(\varphi^+_{f1}(p)) \cap \delta\mathcal{A}_f(\varphi^+_{f2}(p)) = \emptyset$*
5. *for any two responsibilities $\mathbf{G}\ (\forall \vec{x_1}.\varphi_{past1} \supset \varphi^+_{f1}(p)) \in \Phi_{f+}$, $\mathbf{G}\ (\forall \vec{x_2}.\varphi^-_{c2}(p) \supset \varphi_{past2}) \in \Phi_c$,*
   *(a) either for all $\delta$, $\delta\mathcal{A}_f(\varphi^+_{f1}(p)) \cap \delta\mathcal{A}_c(\varphi^-_{c2}(p)) = \emptyset$*
   *(b) or $\varphi_{past1} = \varphi_P$, and for all $P \in \mathcal{A}_c(\varphi^-_{c2}(p))$, for each mgu $\delta$ such that $\delta P \in \delta(\mathcal{A}_f(\varphi^+_{f1}(p)))$, $\delta\varphi_p \vdash \delta\varphi_{past2}$*

*Proof.* By induction on $j$.

Case: $j = 0$

Give any $\widehat{\sigma}_0$ such that,

$$\widehat{\sigma}_0 \mid^{\mathcal{A}}_p = \emptyset, \kappa_0 \supseteq \kappa^p \tag{1}$$

By Lemma D.26,, there exists $a'_0$, $\neg a'_0$, $\tau'_0$ such that

$$G(\widehat{\sigma}_0 \uplus (\emptyset, \emptyset, a'_0, \neg a'_0, \tau'_0, \emptyset), a'_0, \neg a'_0, \tau'_0, i, p, t_0) \text{ holds} \tag{2}$$

and given $\widehat{\sigma}''$ such that $\widehat{\sigma}'' \mid_0 = \emptyset, \widehat{\sigma}'' \uplus \widehat{\sigma}_0 \uplus (\emptyset, \emptyset, a'_0, \neg a'_0, \tau'_0, \emptyset)$ is well-defined

$$\text{implies } V(\widehat{\sigma}'' \uplus \widehat{\sigma}_0 \uplus (\emptyset, \emptyset, a'_0, \neg a'_0, \tau'_0, \emptyset), 0, \Phi) \tag{3}$$

Case: j = k

By I.H. on (k-1),

$$GF'(k-1, \Phi, \{p\}) \tag{1}$$

To show $GF'(k, \Phi, \{p\})$, we unfold $GF'(k-1, \Phi, \{p\})$,
and the first $k-1$ layers of alternating $\forall$ and $\exists$ quantification in $GF'(k, \Phi, \{p\})$,
will be discharged by $GF'(k-1, \Phi, \{p\})$,
now we are obtained, $\forall 0 \leq n < k$,

$$(\widehat{\sigma}_n \,|_{n-1} = \emptyset)(\widehat{\sigma}_n \,|_n = \widehat{\sigma}_n), \mathit{start}(\tau_n) = \tau_0(0), (\widehat{\sigma}_n \,|_p^{\mathcal{A}} = \emptyset), \kappa_n \supseteq \kappa^p \tag{2}$$

$$\forall m \in \mathrm{dom}(a_{pn}), m \geq n, \forall m \in \mathrm{dom}(\neg a_{pn}), m \geq n \tag{3}$$

$$\forall m \in \mathrm{dom}(\tau_{pn}), m \geq n \wedge \mathit{start}(\tau_{pn}) = \tau_0(0) \tag{4}$$

$$\forall P \in \mathrm{range}(a_{pn}), p \text{ is the performer of } P \tag{5}$$

$$\forall P \in \mathrm{range}(\neg a_{pn}), p \text{ is the performer of } P \tag{6}$$

$\forall P \in \mathrm{range}(a_{pn}), \exists\ \mathbf{G}\ (\forall \vec{x}.\varphi_{past} \supset \varphi_f^+(p)) \in \Phi_{f+}, \exists \delta$ such that
$\biguplus_{m=0}^{n} \widehat{\sigma}_m \uplus \biguplus_{m=0}^{n} (\emptyset, \emptyset, a'_m, \neg a'_m, \tau'_m, \emptyset), n \vDash \delta\varphi_{past}$,
and $\biguplus_{m=0}^{n} \widehat{\sigma}_m \uplus \biguplus_{m=0}^{n} (\emptyset, \emptyset, a'_m, \neg a'_m, \tau'_m, \emptyset), n \Vdash \delta\varphi_f^+$

$$\exists P' \in \mathcal{A}_f(\delta\varphi_f^+(p)),\ \text{and } \exists \delta^0,\ \text{and } P = \delta^0 P' \tag{7}$$

$$\forall P \in \mathrm{range}(\neg a_{pn}(i)) \exists P' \in \mathcal{A}_f(\varphi_f^-(p)) \cup \mathcal{A}_c(\varphi_c^-(p)) \exists \delta^0,\ \text{and } \delta^0 P = \delta^0 P' \tag{8}$$

$$\forall m \in \mathrm{dom}(\neg a_{pn}) \text{ and } m > n, \forall P \in a_{pn}(m) \exists P' \in \mathcal{A}_f(\varphi_f^-(p)), \exists \delta^0,\ \text{and } \delta^0 P = \delta^0 P' \tag{9}$$

given any $\widehat{\sigma}'$ such that $\widehat{\sigma}' \,|_n = \emptyset$, $\forall \varphi \in \Phi$,

$$\widehat{\sigma}' \uplus \widehat{\sigma}_0 \uplus (\emptyset, \emptyset, a_{p0}, \neg a_{p0}, \tau_{p0}, \emptyset) \uplus \cdots \uplus \widehat{\sigma}_n \uplus (\emptyset, \emptyset, a_{pn}, \neg a_{pn}, \tau_{pn}, \emptyset), n \Vdash \varphi \tag{10}$$

Give any $\widehat{\sigma}_k$, such that

$$(\widehat{\sigma}_k \,|_{k-1} = \emptyset)(\widehat{\sigma}_k \,|_k = \widehat{\sigma}_k), (\widehat{\sigma}_k \,|_p^{\mathcal{A}} = \emptyset), \mathit{start}(\tau_k) = \tau_0(0), \kappa_k \supseteq \kappa^p \tag{11}$$

Let $\widehat{\sigma} = \widehat{\sigma}_0 \uplus (\emptyset, \emptyset, a'_0, \neg a'_0, \tau'_0, \emptyset) \uplus \cdots \uplus \widehat{\sigma}_k$,

By Lemma D.12, Lemma D.15, Lemma D.17, Lemma D.2, (7) ,

$\forall P \in \mathrm{range}(a_{pn}), \exists\ \mathbf{G}\ (\forall \vec{x}.\varphi_{past} \supset \varphi_f^+(p)) \in \Phi_{f+}, \exists \delta$ such that
$\widehat{\sigma}, n \vDash \delta\varphi_{past}$, and $\widehat{\sigma}, n \Vdash \delta\varphi_f^+$

$$\exists P' \in \mathcal{A}_f(\delta\varphi_f^+(p)),\ \text{and } \exists \delta^0,\ \text{and } P = \delta^0 P' \tag{12}$$

By Lemma D.26, for all exists $a'_k$, $\neg a'_k$, $\tau'_k$ such that

$$G(\widehat{\sigma}(\emptyset, \emptyset, a'_k, \neg a'_k, \tau'_k, \emptyset), a'_k, \neg a'_k, \tau'_k, i, p, \tau_0(0)) \text{ holds} \tag{13}$$

and given $\widehat{\sigma}''$ such that $\widehat{\sigma}'' \,|_k = \emptyset, \widehat{\sigma}'' \uplus \widehat{\sigma} \uplus (\emptyset, \emptyset, a'_k, \neg a'_k, \tau'_k, \emptyset)$ is well-defined

$$\text{implies } V(\widehat{\sigma}'' \uplus \widehat{\sigma} \uplus (\emptyset, \emptyset, a'_k, \neg a'_k, \tau'_k, \emptyset), k, \Phi) \tag{14}$$

□

**Theorem D.28** (Feasibility compositions for one agent)**.** *Let $\Phi$ be a set of responsibilities, and $\Phi = \Phi_c, \Phi_{f+}, \Phi_{f-}$ where $\forall\ \mathbf{G}\ \varphi \in \Phi_c$, $\varphi$ is of the form $\forall \vec{x}.\varphi_c^-(p) \supset \varphi_{past}$, $\forall\ \mathbf{G}\ \varphi \in \Phi_{f+}$, $\varphi$ is of the form $\forall \vec{x}.\varphi_{past} \supset \varphi_f^+(p)$, and $\forall\ \mathbf{G}\ \varphi \in \Phi_{f-}$, $\varphi$ is of the form $\forall \vec{x}.\varphi_{past} \supset \varphi_f^-(p)$.*
*$\Phi$ is feasible for agent $p$ if*

1. *for all $\mathbf{G}\ (\forall \vec{x}.\varphi_{past} \supset \varphi_f^-(p)) \in \Phi_{f-}$, $\cdot;\cdot \vdash \varphi_f^-$ $\mathsf{sat}$*

2. *for all $\mathbf{G}\ (\forall \vec{x}.\varphi_{past} \supset \varphi_f^+(p)) \in \Phi_{f+}$, $\vdash \varphi_{past}$ $\mathsf{fin}$, $\cdot;\cdot \vdash \varphi_f^+$ and $\mathcal{A}_f(\varphi_f^+(p)) \vdash \varphi_{past}$*

3. *for any two responsibilities $\mathbf{G}\ (\forall \vec{x_1}.\varphi_{past1} \supset \varphi_{f1}^+(p)) \in \Phi_{f+}$, $\mathbf{G}\ (\forall \vec{x_2}.\varphi_{past2} \supset \varphi_{f2}^+(p)) \in \Phi_{f+}$, $\mathcal{A}_f(\varphi_{fj}^+(p)) \vdash \varphi_{pasti}$ $(i = 1, 2,\ j = 1, 2,\ i \neq j)$*

4. *for any two responsibilities $\mathbf{G}\ (\forall \vec{x_1}.\varphi_{past1} \supset \varphi_{f1}^+(p)) \in \Phi_{f+}$, $\mathbf{G}\ (\forall \vec{x_2}.\varphi_{past2} \supset \varphi_{f2}^-(p)) \in \Phi_{f-}$, for all $\delta$, $\delta\mathcal{A}_f(\varphi_{f1}^+(p)) \cap \delta\mathcal{A}_f(\varphi_{f2}^+(p)) = \emptyset$*

5. *for any two responsibilities $\mathbf{G}\ (\forall \vec{x_1}.\varphi_{past1} \supset \varphi_{f1}^+(p)) \in \Phi_{f+}$, $\mathbf{G}\ (\forall \vec{x_2}.\varphi_{c2}^-(p) \supset \varphi_{past2}) \in \Phi_c$,*

   *(a) either for all $\delta$, $\delta\mathcal{A}_f(\varphi_{f1}^+(p)) \cap \delta\mathcal{A}_c(\varphi_{c2}^-(p)) = \emptyset$*

(b) or $\varphi_{past1} = \varphi_P$, and for all $P \in \mathcal{A}_c(\varphi^-_{c2}(p))$, for each mgu $\delta$ such that $\delta P \in \delta(\mathcal{A}_f(\varphi^+_{f1}(p)))$, $\delta\varphi_p \vdash \delta\varphi_{past2}$

*Proof.* By Lemma D.27. □

**Lemma D.29.** *For all $\widehat{\sigma}$, $i$ and $\widehat{\sigma}'$ such that $\widehat{\sigma} \uplus \widehat{\sigma}'$ is well-defined $\widehat{\sigma}' \mid_{i-1} = \emptyset$, and $\forall P \in range(a') \cup range(\neg a')$, $p$ is not a performer, if $\widehat{\sigma}, i \Vdash \varphi_i$ where $\varphi_i$ is $\forall \vec{x}.\varphi^-_c(p) \supset \varphi_{past}$ or $\forall \vec{x}.\varphi_{past} \supset \varphi^+_f(p)$, or $\forall \vec{x}.\varphi_{past} \supset \varphi^-_f(p)$, and $p \vdash \varphi_{past}$ StrictPast then $\widehat{\sigma} \uplus \widehat{\sigma}', i \Vdash \varphi_i$*

**Proof** (sketch): By Lemma D.12, Lemma D.15, Lemma D.17, Lemma D.5, □

**Lemma D.30** (Strong feasibility composition for multiple agents)**.** *Given a group of agents $Sa$, let $\Phi_p$ be the set of responsibilities for each agent $p \in Sa$. $\Phi_p = \Phi^p_c, \Phi^p_{f+}, \Phi^p_{f-}$ where $\forall$ $\mathbf{G}$ $\varphi \in \Phi^p_c$, $\varphi$ is of the form $\forall \vec{x}.\varphi^-_c(p) \supset \varphi_{past}$, $\forall$ $\mathbf{G}$ $\varphi \in \Phi^p_{f+}$, $\varphi$ is of the form $\forall \vec{x}.\varphi_{past} \supset \varphi^+_f(p)$, and $\forall$ $\mathbf{G}$ $\varphi \in \Phi^p_{f-}$, $\varphi$ is of the form $\forall \vec{x}.\varphi_{past} \supset \varphi^-_f(p)$.*

$$
\begin{aligned}
\textit{Let } G(\widehat{\sigma}, a, \neg a, \tau, i, Sa) = {} & \forall k \in \mathrm{dom}(a), k \geq i \wedge \forall k \in \mathrm{dom}(\neg a), k \geq i \wedge \forall k \in \mathrm{dom}(\tau), k \geq i \wedge \mathit{start}(\tau) = t_0 \wedge \\
& \forall P \in \mathit{range}(a),\ \exists p \in Sa, \text{ such that } p \text{ is the performer of } P \wedge \\
& \forall P \in \mathit{range}(\neg a),\ \exists p \in Sa, \text{ such that } p \text{ is the performer of } P \wedge \\
& \forall k \in \mathrm{dom}(a), \forall P \in a(k),\ \exists p \in Sa, \\
& \quad \exists\ \mathbf{G}\ \varphi_i \in \Phi_{f+},\ \exists \delta \text{ such that } \widehat{\sigma}, i \vDash \delta\varphi_{past}, \text{ and} \\
& \quad \widehat{\sigma}, i \Vdash \delta\varphi^+_f\ \exists P' \in \mathcal{A}_f(\delta\varphi^+_f(p)), \text{ and } \exists \delta^0, \text{ and } P = \delta^0 P' \\
& \quad \forall P \in (\neg a(k)),\ \exists p \in Sa,\ \exists P' \in \mathcal{A}_f(\varphi^-_f(p)) \cup \mathcal{A}_c(\varphi^-_c(p)),\ \exists \delta^0, \text{ and } \delta^0 P = \delta^0 P' \\
& \forall k \in \mathrm{dom}(\neg a), k > i, \forall P \in a(k),\ \exists p \in Sa,\ \exists P' \in \mathcal{A}_f(\varphi^-_f(p)),\ \exists \delta^0, \text{ and } \delta^0 P = \delta^0 P'. \\
\textit{Let } V(\widehat{\sigma}, i, \Phi) = {} & \forall\ \mathbf{G}\ \varphi_i \in \Phi,\ \mathit{Tr}(\widehat{\sigma}), i \Vdash \varphi_i
\end{aligned}
$$

*For all $j$, $GF'(j, \bigcup_p \Phi_p, Sa)$, if for each $\mathbf{G}$ $\varphi_i \in \Phi$, the past formula in $\varphi_i$ is $\varphi_{past}$, and $p \vdash \varphi_{past}$ StrictPast, and*

1. *for all $\mathbf{G}$ $(\forall \vec{x}.\varphi_{past} \supset \varphi^-_f(p)) \in \Phi_{f-}$, $\cdot;\cdot \vdash \varphi^-_f$ sat*
2. *for all $\mathbf{G}$ $(\forall \vec{x}.\varphi_{past} \supset \varphi^+_f(p)) \in \Phi_{f+}$, $\vdash \varphi_{past}$ fin, $\cdot;\cdot \vdash \varphi^+_f$ and $\mathcal{A}_f(\varphi^+_f(p)) \vdash \varphi_{past}$*
3. *for any two responsibilities $\mathbf{G}$ $(\forall \vec{x_1}.\varphi_{past1} \supset \varphi^+_{f1}(p)) \in \Phi_{f+}$, $\mathbf{G}$ $(\forall \vec{x_2}.\varphi_{past2} \supset \varphi^+_{f2}(p)) \in \Phi_{f+}$, $\mathcal{A}_f(\varphi^+_{fj}(p)) \vdash \varphi_{pasti}$ $(i = 1, 2,\ j = 1, 2,\ i \neq j)$*
4. *for any two responsibilities $\mathbf{G}$ $(\forall \vec{x_1}.\varphi_{past1} \supset \varphi^+_{f1}(p)) \in \Phi_{f+}$, $\mathbf{G}$ $(\forall \vec{x_2}.\varphi_{past2} \supset \varphi^-_{f2}(p)) \in \Phi_{f-}$, for all $\delta$, $\delta\mathcal{A}_f(\varphi^+_{f1}(p)) \cap \delta\mathcal{A}_f(\varphi^+_{f2}(p)) = \emptyset$*
5. *for any two responsibilities $\mathbf{G}$ $(\forall \vec{x_1}.\varphi_{past1} \supset \varphi^+_{f1}(p)) \in \Phi_{f+}$, $\mathbf{G}$ $(\forall \vec{x_2}.\varphi^-_{c2}(p) \supset \varphi_{past2}) \in \Phi_c$,*
   - (a) *either for all $\delta$, $\delta\mathcal{A}_f(\varphi^+_{f1}(p)) \cap \delta\mathcal{A}_c(\varphi^-_{c2}(p)) = \emptyset$*
   - (b) *or $\varphi_{past1} = \varphi_P$, and for all $P \in \mathcal{A}_c(\varphi^-_{c2}(p))$, for each mgu $\delta$ such that $\delta P \in \delta(\mathcal{A}_f(\varphi^+_{f1}(p)))$, $\delta\varphi_p \vdash \delta\varphi_{past2}$*

*Proof.* By induction on $j$. Similar to the proof of Lemma D.27. We use Lemma D.26 to obtain planned trace for each agent, then we use Lemma D.29 to compose traces from different agents.

□

**Theorem D.31** (Feasibility composition for multiple agents)**.** *Given a group of agents $Sa$, let $\Phi_p$ be the set of responsibilities for each agent $p \in Sa$. $\Phi_p = \Phi^p_c, \Phi^p_{f+}, \Phi^p_{f-}$ where $\forall$ $\mathbf{G}$ $\varphi \in \Phi^p_c$, $\varphi$ is of the form $\forall \vec{x}.\varphi^-_c(p) \supset \varphi_{past}$, $\forall$ $\mathbf{G}$ $\varphi \in \Phi^p_{f+}$, $\varphi$ is of the form $\forall \vec{x}.\varphi_{past} \supset \varphi^+_f(p)$, and $\forall$ $\mathbf{G}$ $\varphi \in \Phi^p_{f-}$, $\varphi$ is of the form $\forall \vec{x}.\varphi_{past} \supset \varphi^-_f(p)$.*

*The union of $\Phi_p$ for all $p \in Sa$ is feasible for $Sa$. if for each $\mathbf{G}$ $\varphi_i \in \Phi$, the past formula in $\varphi_i$ is $\varphi_{past}$, and $p \vdash \varphi_{past}$ StrictPast, and*

1. *for all* $\mathbf{G}\ (\forall \vec{x}.\varphi_{past} \supset \varphi_f^-(p)) \in \Phi_{f-}$, $\cdot;\cdot \vdash \varphi_f^-\ \mathsf{sat}$

2. *for all* $\mathbf{G}\ (\forall \vec{x}.\varphi_{past} \supset \varphi_f^+(p)) \in \Phi_{f+}$, $\vdash \varphi_{past}\ \mathsf{fin}$, $\cdot;\cdot \vdash \varphi_f^+$ *and* $\mathcal{A}_f(\varphi_f^+(p)) \vdash \varphi_{past}$

3. *for any two responsibilities* $\mathbf{G}\ (\forall \vec{x_1}.\varphi_{past1} \supset \varphi_{f1}^+(p)) \in \Phi_{f+}$, $\mathbf{G}\ (\forall \vec{x_2}.\varphi_{past2} \supset \varphi_{f2}^+(p)) \in \Phi_{f+}$, $\mathcal{A}_f(\varphi_{fj}^+(p)) \vdash \varphi_{pasti}$ $(i = 1, 2,\ j = 1, 2,\ i \neq j)$

4. *for any two responsibilities* $\mathbf{G}\ (\forall \vec{x_1}.\varphi_{past1} \supset \varphi_{f1}^+(p)) \in \Phi_{f+}$, $\mathbf{G}\ (\forall \vec{x_2}.\varphi_{past2} \supset \varphi_{f2}^-(p)) \in \Phi_{f-}$, *for all* $\delta$, $\delta\mathcal{A}_f(\varphi_{f1}^+(p)) \cap \delta\mathcal{A}_f(\varphi_{f2}^+(p)) = \emptyset$

5. *for any two responsibilities* $\mathbf{G}\ (\forall \vec{x_1}.\varphi_{past1} \supset \varphi_{f1}^+(p)) \in \Phi_{f+}$, $\mathbf{G}\ (\forall \vec{x_2}.\varphi_{c2}^-(p) \supset \varphi_{past2}) \in \Phi_c$,

   (a) *either for all* $\delta$, $\delta\mathcal{A}_f(\varphi_{f1}^+(p)) \cap \delta\mathcal{A}_c(\varphi_{c2}^-(p)) = \emptyset$

   (b) *or* $\varphi_{past1} = \varphi_P$, *and for all* $P \in \mathcal{A}_c(\varphi_{c2}^-(p))$, *for each mgu* $\delta$ *such that* $\delta P \in \delta(\mathcal{A}_f(\varphi_{f1}^+(p)))$, $\delta\varphi_p \vdash \delta\varphi_{past2}$

*Proof.* By Lemma D.30. $\square$