A Sequent Calculus for Counterfactual Reasoning

McKenna McCall
Carnegie Mellon University
USA

Lay Kuan Loh
Carnegie Mellon University
USA

Limin Jia
Carnegie Mellon University
USA

ABSTRACT

Counterfactual conditions such as "if A were not true, then C would not have been true" have been formally studied by philosophers for causal claims for decades. Counterfactuals are often used informally in practice for diagnosing systems and identifying errors or misconfigurations. This paper develops a proof theory for counterfactual reasoning of Horn clauses, which have applications in domains including security and database and program analysis. The application to security that this paper focuses on is modeling and reasoning about probing attacks in Datalog-based trust management systems, where an attacker can apply counterfactual reasoning to obtain sensitive information embedded in the system.

Our work is inspired by a Hilbert-style axiomatized system for counterfactual reasoning for Horn clauses, which are hard to use to construct proofs or study properties of the system. To alleviate this difficulty, we develop a sequent calculus from first principles. We show that the sequent calculus has cut elimination and is sound and complete with regard to the corresponding Hilbert style axiomatized system. We also show how to construct proofs that model practical counterfactual reasoning scenarios in trust management systems using our sequent calculus rules.

KEYWORDS

counterfactual reasoning, access control, Horn clauses, sequent calculus, cut elimination

1 INTRODUCTION

Counterfactual conditions such as "if A were true, then C would have been true" and "if A were not true, then C would not have been true" describe what could or would happen under different conditions. For decades, philosophers have investigated using counterfactuals to explain causal claims [10, 22, 26]. In the above example, A is a likely cause of C. Counterfactual reasoning is a meta-level reasoning about a system. In practice, people often apply ad-hoc counterfactual reasoning to diagnosing and analyzing concrete systems, both for legitimate purposes, such as debugging, as well as malicious ones, such as probing secrets embedded in the system. The analyst crafts a set of inputs, feeds each input to the system, and then draws conclusions based on differences in observable outputs. Counterfactuals such as "if the input weren't a null string, the system would not have crashed" are used to pinpoint possible causes of problems.

One important research question is how to formalize counterfactual reasoning to provide a solid foundation for correct uses of such reasoning in practice. In particular, we are interested in formalizing counterfactual reasoning in systems that can be modeled using Horn clauses. One concrete example of such systems are Datalog-based trust management systems [4, 12]. These systems typically use a set of Horn clauses to specify the attributes of users and the policies for who can access what system resources. For instance, suppose University U has a policy stating that a student can access the gym as long as the student signs up for the gym service. We can encode this policy as the following Horn clause:

$$U.\mathsf{canAccessGym}(x) \text{ :- } x.\mathsf{registerForGym}.$$

Here, :- is read as a reversed implication. We write $K.\varphi$ to mean that principal K asserts that φ is true. U is the principal representing the authority of the university. Note that this notation can be encoded as a predicate (Section 2).

One type of counterfactual reasoning in such systems is a *probing attack*. A probing attack against a trust management system involves someone who has the ability to submit additional clauses to the system. The attacker probes the system by adding clauses to the system and observing changes in access rights. The added clauses often make use of sensitive information of the system that the attacker should not have access to.

Continuing the example, an attacker A wants to know whether her friend B has a low GPA, information which is private and should not be leaked to A. A knows that she does not have access to the gym currently. A then adds the following clause into the system:

$$A.\mathsf{registerForGym} \text{ :- } U.\mathsf{lowGPA(B)}.$$

This clause states that if the University authority says that student B has a low GPA, then A registers for the gym. Given this additional clause, A gains access to the gym. From these two cases, A can deduce that it must be the case that B has a low GPA. The attacker is sure of the secret value based on counterfactual reasoning: if the secret value were different, then a particular access right would not have been granted.

This paper aims to design a formal logical system for counterfactual reasoning of Horn clauses from first principles. Existing work on counterfactual reasoning of Horn clauses provides axiomatized formulations. The advantage of a sequent calculus over a Hilbert-style axiomatized system is that it is amenable to automated theorem proving. If the sequent calculus has subformula properties, i.e., formulas in subderivations are subformulas of the conclusion, then sequent rules can be applied exhaustively to construct proofs. Axiomatized systems rely more on clever applications of axioms, so proofs can sometimes be hard to construct. Sequent calculus rules also provide an inductive principle for proving meta-properties of

PLAS'17, October 30, 2017, Dallas, TX, USA

ACM ISBN 978-1-4503-5099-0/17/10…$15.00
https://doi.org/10.1145/3139337.3139342

the logical system. Building on prior work of a Hilbert-style axiomatized system for counterfactual reasoning for trust management policies [5], we define a sequent calculus.

We choose Horn clauses to be the subject of counterfactual reasoning because Horn clauses, due to their simplicity, have been widely used in many areas, including databases [1], logic programming [37], network verification [14, 24], program analysis [8, 38], and security [4, 7, 12]. Therefore, a solid foundation for counterfactual reasoning for Horn clauses could have impact in debugging, diagnosis, and identifying attacks in all of the above areas.

This paper makes the following technical contributions.

- We define the first sequent calculus for counterfactual reasoning of Horn clauses.
- We prove that the sequent calculus has cut elimination and that it is sound and complete with regard to an axiomatized system.
- We show how to construct proofs that demonstrate probing attacks using our sequent calculus rules.

The rest of this paper is organized as follows. We briefly review Horn clauses and trust management systems in Section 2. We present the Hilbert-style system in Section 3. Our sequent calculus rules are explained in Section 4 and properties of the calculus are discussed in Section 5. We discuss design choices and broader applications of this formalism in Section 6. Related work is presented in Section 7. Proof details may be found in our companion technical report [25].

2 BACKGROUND

A Horn clause is a first-order logical formula of the form $p_1 \wedge p_2 \cdots \wedge p_n \rightarrow q$, where each p_i and q are atomic predicates. Variables in the formula are quantified at the outermost level. We will use the Prolog/Datalog notation and write: q :- $p_1, \cdots, p_n$. We call p_1 to p_n, formulas to the right of :-, the body of the clause, and q the head of the clause. When the domain of the variables is finite, first-order clauses can be instantiated to a finite set of propositional clauses. We write w to denote a set of propositional clauses. We write $w \Vdash_d p$ to mean that p is derivable from w, defined using the inference rules shown in Figure 1. p is derivable from a policy world if either p is in the policy world, or p is the head of a clause in the policy world, and all body propositions of that rule are derivable from the same policy world.

A trust management system can be viewed as a framework for managing distributed access control [6]. Horn clauses have been used for specifying policies in trust management systems [4, 12, 20, 23]. Below is an example encoding a Unix file system access-control policy, stating that the owner of a file can open that file and that A owns the file *tmp*.

$$S.\mathsf{owns}(A, tmp). \qquad S.\mathsf{mayOpen}(x, f) \text{:-} S.\mathsf{owns}(x, f).$$

Recall from the example earlier, $S.p$ means principal S says p. The says connective was introduced in authorization logics, whose semantics and proof theory have been studied in depth (e.g., [9, 13, 15, 16]). We do not aim to model any specific authorization logic; our goal is to use the syntax to encode real-world access control examples. The says connective can be encoded by making the principal S the first argument of the predicate. Below is the pure Horn clause encoding.

$$\mathsf{owns}(S, A, tmp). \qquad \mathsf{mayOpen}(S, x, f) \text{:-} \mathsf{owns}(S, x, f).$$

$$\frac{}{w, p \Vdash_d p}\ \textsc{At} \qquad \frac{(w, p \text{ :- } q_1, \cdots, q_n \Vdash_d q_i)_{i \in [1,n]}}{w, p \text{ :- } q_1, \cdots, q_n \Vdash_d p}\ \textsc{Dlog}$$

Figure 1: Derivability relation for Datalog rules

Role-based and attribute-based access control policies can also be encoded as a set of Horn clauses specifying the role, role assignments, attributes, and access control decisions based on roles and attributes. For the rest of this paper, we will use access control policies as our main application domain. We will use policies, rules, and clauses interchangeably. We call w a policy context.

When a principal requests access to a system resource, a proof that the principal is allowed access needs to be constructed using the clauses that encode the access control policies. From the above two policies, we can prove that A can open file *tmp*, using the proof system composed of the rules in Figure 1.

In trust management systems, the Horn clauses that encode the access control policies and related facts are also called *credentials*, as some systems require that they are signed using the private keys of authorities. For instance, $\mathsf{owns}(S, A, tmp)$ is stored as a digital certificate signed by S's private key.

Trust management systems are distributed; principals can issue their own credentials and are allowed to make their own policies. The probing attack discussed earlier is caused by the attacker issuing credentials that make use of facts that are supposed to be secret to the attacker. Based on differences in observable effects (i.e., proofs of accesses), the attacker can apply counterfactual reasoning and deduce the existence or absence of sensitive credentials.

3 COUNTERFACTUAL REASONING

We review the formalism for specifying and reasoning about counterfactuals of Horn clauses introduced by Becker et al. [5]. Our formalism uses the same syntax and semantics, but a different proof system (Section 4).

3.1 Syntax and Semantics of Counterfactual Conditions

In the example from Section 1, the probes that the attacker attempts are counterfactual conditions: if these policies were not true, access would not have been granted. Becker et al. internalized counterfactual conditions using an indexed modal operator: $\Box_w \varphi$. Informally, $\Box_w \varphi$ is true if after submitting policies in w, φ is derivable. For instance, $\Box_{b \text{ :- } a} b$ means after submitting the clause b :- a, b can be derived. We summarize the syntax of all the logical formulas below.

Policy	$\gamma ::= \top \mid p \mid q \text{ :- } \vec{p} \mid \gamma \wedge \gamma'$
Policy Context	$w ::= \top \mid w, p \mid w, q \text{ :- } \vec{p}$
Formula	$\varphi ::= \gamma \mid \bot \mid \neg\varphi \mid \varphi \wedge \varphi' \mid \varphi \vee \varphi'$
	$\mid \ \varphi \rightarrow \varphi' \mid \Box_\gamma \varphi$

A policy, denoted γ, can be an atomic proposition, a Horn clause, or the conjunction of a set of Horn clauses. Formulas, denoted φ, include all of the connectives from propositional logic. The main addition is $\Box_\gamma \varphi$. It means that φ can be derived when the policy

$$
\begin{array}{lll}
w \Vdash p & \text{iff} & w \Vdash_d p \\
w \Vdash \top & \text{iff} & \text{always} \\
w \Vdash \bot & \text{iff} & \text{never} \\
w \Vdash \neg\varphi & \text{iff} & w \nVdash \varphi \\
w \Vdash \varphi \wedge \varphi' & \text{iff} & w \Vdash \varphi \text{ and } w \Vdash \varphi' \\
w \Vdash \varphi \vee \varphi' & \text{iff} & w \Vdash \varphi \text{ or } w \Vdash \varphi' \\
w \Vdash \varphi \rightarrow \varphi' & \text{iff} & \text{if } w \Vdash \varphi \text{ then } w \Vdash \varphi' \\
w \Vdash \Box_{\gamma'} \varphi & \text{iff} & w, \overline{\gamma'} \Vdash \varphi
\end{array}
$$

Figure 2: Formula Semantics

γ is submitted. The Horn clauses are themselves counterfactual conditions as well: p :- $p_1, \ldots, p_n$ is equivalent to $\Box_{(\wedge_{i=1}^{n} p_i)}\, p$. We will use these two notations interchangeably. A policy context w is a set of Horn clauses. The empty context is $\top$. We write $\wedge(w)$ to denote the policy that is the conjunction of every clause in w, and $\overline{\gamma}$ to denote the set of clauses in γ.

The example in Section 1 can be formalized as follows. φ_1 encodes A's first probe: A cannot access the gym. φ_2 is A's second probe: after submitting the policy "if B has a low GPA, then A registers for the gym", A can access the gym. γ_0 encodes the system's policy: anyone who registers for the gym can access it. γ_0 is the policy context under which the probes are carried out. As will be clear after we formally define probing attacks in Section 3.2, the attacker need not know γ_0.

$$
\begin{array}{lcl}
\varphi_1 & = & \neg U.\text{canAccessGym(A)} \\
\varphi_2 & = & \Box_{A.\text{registerForGym :- } U.\text{lowGPA(B)}}\, U.\text{canAccesssGym(A)} \\
\gamma_0 & = & U.\text{canAccessGym}(x) \text{ :- } x.\text{registerForGym}
\end{array}
$$

A proof system (see Section 3.3) for counterfactual reasoning can show that $U.\text{lowGPA}(B)$ is the logical consequence of the conjunction of φ_1 and φ_2.

Key definitions for the semantics of formulas are summarized in Figure 2. Formulas are interpreted under a set of policies w. The semantics for the atomic predicate is defined in terms of the derivability of Horn clauses. The counterfactual condition $\Box_{\gamma'} \varphi$ is true under policies w if φ can be derived from the union of w and γ'. The rest of the connectives are standard. We say that a formula φ is *valid* if $\forall w, w \Vdash \varphi$. We write $\Vdash \varphi$ to mean that φ is valid.

It is important to note that, while Horn clauses may be read as reverse implication, q :- p is not logically equivalent to $p \rightarrow q$. Consider the following example. Let $\gamma = \top$. Then, $\gamma \Vdash p \rightarrow q$ since $p \rightarrow q$ is logically equivalent to $\neg p \vee q$ and $\gamma \nVdash p$. But, $\gamma \nVdash \Box_p\, q$ because $\top, p \nVdash q$. The counterfactual statement $\Box_p\, q$ indicates a stronger relation between p and q than the material implication $p \rightarrow q$, so they cannot be used interchangeably.

3.2 Formal Definitions of Probing Attack

Probing attacks can be formally defined based on the semantics shown in Figure 2. A *probe* π is a formula of the form $\Box_\gamma\, \varphi$, where φ is a $\Box$-free formula. $\overline{\gamma}$ is called *the probe credential set* and φ is called the *probe query*. An observation of a probe π under a set of policies w_0 is either π if $w_0 \Vdash \pi$, or $\neg\pi$ otherwise. In our example, $U.\text{canAccesssGym(A)}$ is a probe query and a probe credential is $A.\text{registerForGym :- } U.\text{lowGPA(B)}$. A probing attack on w_0 consisting of probes $\{\pi_1, \ldots, \pi_n\}$ is the conjunction of the observations of $\pi_i \in \{\pi_1, \ldots, \pi_n\}$ under w_0. We treat trust management systems as black boxes, so w_0 is not visible to an attacker, only the observations under w_0. In our example, w_0 is the set of U's policies, including γ_0 defined above. The probe is the conjunction of φ_1 and φ_2. If φ is a probing attack on w_0, then the attacker knows that $w_0 \Vdash \varphi$. If some property φ' holds in all w such that $w \Vdash \varphi$, then the attacker knows for sure that φ' holds under w_0. We say that φ' is *detectable*.

Definition 1 (Detectability)**.** A formula φ' is detectable in a probing attack φ iff $\forall w$ s.t. $w \Vdash \varphi$, $w \Vdash \varphi'$

Conversely, if there exists some policy w' such that φ holds in w', but $w' \nVdash \varphi'$, then attacker cannot be certain that φ' holds given the results of the probes. Thus, φ' is said to be opaque.

Definition 2 (Opacity)**.** A formula φ' is opaque in a probing attack φ iff it is not detectable in φ, or equivalently, $\exists w_1, w_1 \Vdash \varphi$ and $w_1 \Vdash \varphi'$ and $\exists w_2, w_2 \Vdash \varphi$ and $w_2 \nVdash \varphi'$

Note that our definition of Opacity differs slightly from that of Becker et al. We made explicit that there exist two policy contexts that produce the same observation for the probe φ, but differ in whether φ' is observable.

Lemma 3 (Probing attacks)**.** *A formula φ' is detectable in a probing attack φ iff $\forall w$, $w \Vdash \varphi \rightarrow \varphi'$*

In our example, $\forall w$, $w \Vdash \varphi_1 \wedge \varphi_2$ implies $w \Vdash \text{U.lowGPA}(B)$. Therefore, $\text{U.lowGPA}(B)$ is detectable in a probing attack consisting of φ_1 and φ_2 (see Section 3.1).

Consider a similar probing attack on a different policy, γ_0', also not visible to A, which states that a person can access the gym if they have registered for the gym and paid their fees. The two probes φ_1 and φ_2' are shown below.

$$
\begin{array}{lcl}
\gamma_0' & = & U.\text{canAccessGym}(x) \text{ :- } (x.\text{registerForGym}, x.\text{paidFees}) \\
\varphi_1 & = & \neg U.\text{canAccessGym(A)} \\
\varphi_2' & = & \Box_{A.\text{registerForGym :- } U.\text{lowGPA(B)}}\, \neg U.\text{canAccesssGym(A)}
\end{array}
$$

In this example, A sees that they cannot access the gym (φ_1), so they input the credentials: "if B has a low GPA, A registers for the gym" and observes that they still cannot access the gym (φ_2'). From this information, A may want to conclude that B does not have a low GPA ($\neg\text{U.lowGPA}(B)$), but it is easy to construct a w_1 and w_2 to show that $\neg\text{U.lowGPA}(B)$ is opaque given the above two probes. If we let $w_1 = \gamma_0'$, we find that $w_1 \Vdash \varphi_1 \wedge \varphi_2'$ and $w_1 \Vdash \neg\text{U.lowGPA}(B)$. If we let $w_2 = (\text{U.lowGPA}(B), \gamma_0')$, then, $w_2 \Vdash \varphi_1 \wedge \varphi_2'$, but $w_2 \nVdash \neg\text{U.lowGPA}(B)$, therefore $\neg\text{U.lowGPA}(B)$ is opaque in the probing attack $\varphi_1 \wedge \varphi_2'$.

3.3 Proof System

Checking whether a probing attack can detect a secret requires us to check $\forall w, w \Vdash \varphi \rightarrow \varphi'$. This is inconvenient, as we need to examine all possible w. A proof system with derivation rules is helpful in such situations. Becker et al. [5] provides an axiomatization of counterfactual reasoning for Horn clauses (summarized in Figure 3). Here, $\vdash \varphi$ denotes φ is provable by those axioms and proof rules. It has been shown that $\vdash \varphi$ iff $\Vdash \varphi$. Therefore, showing φ' is detectable in probing attack φ is reduced to constructing a proof that $\vdash \varphi \rightarrow \varphi'$.

Next, we explain the axioms and proof rules in more detail. Axioms (Cl1)-(Cl3) and Modus Ponens (MP) are from classical propositional logic.

Axioms

$\vdash \varphi \to \varphi' \to \varphi$ (Cl1)
$\vdash (\varphi \to \varphi' \to \varphi'') \to (\varphi \to \varphi') \to \varphi \to \varphi''$ (Cl2)
$\vdash (\neg\varphi \to \neg\varphi') \to \varphi' \to \varphi$ (Cl3)
$\vdash \Box_\gamma\ (\varphi \to \varphi') \to \Box_\gamma\ \varphi \to \Box_\gamma\ \varphi'$ (K)
$\vdash \Box_\gamma\ \gamma$ (C1)
$\vdash \Box_\gamma\ \varphi \to \gamma \to \varphi$ (C2)
$\vdash \Box_{(p\ :\text{-}\ \vec{q})}\ \varphi \to (\vec{q} \to p) \to \varphi$ (Dlog)
(provided φ is $\Box$-free)
$\vdash \Box_\gamma\ \neg\varphi \leftrightarrow \neg\Box_\gamma\ \varphi$ (Fun)
$\vdash \Box_{\gamma\wedge\gamma'}\ \varphi \leftrightarrow \Box_\gamma\ \Box_{\gamma'}\ \varphi$ (Perm)

Proof rules

If $\vdash \varphi$ and $\vdash \varphi \to \varphi'$, then $\vdash \varphi'$ (MP)
If $\vdash \varphi$ then $\vdash \Box_\gamma\ \varphi$ (N)
If $\vdash \gamma \to \gamma'$ and φ is $\neg$ free (Mon)
then $\vdash \Box_{\gamma'}\ \varphi \to \Box_\gamma\ \varphi$

Figure 3: Axiomatized Proof System

Axiom (K) states that Modus Ponens is preserved under counterfactual conditions. Rule (N) states that if φ is true, then φ is true given additional policies.

Axiom (C1) is a basic requirement for counterfactuals, that if γ is the case, then γ would hold. Axiom (C2) states that counterfactual conditionals are stronger than material implication.

Axiom (Dlog) is the key counterfactual argument for Horn clauses. It captures the essence of Horn clause reasoning. The left hand side of the implication means that "φ would hold in the policy if the clause p :- $\vec{q}$ were submitted". The right hand side of the implication, can be expanded as $(\vec{q} \wedge \neg p) \vee \varphi$. This means that the left hand side holds only if: either (1) φ holds in the policy anyway, even without submitting p :- $\vec{q}$, or (2) the clause must be crucial for making φ true. This is only possible if the body of the clause ($\vec{q}$) are all derivable in the policy, and furthermore that p does not already hold in the policy. Otherwise, the clause could not possibly be crucial.

Axiom (Dlog) only holds for $\Box$-free φ. Intuitively, if additional clauses can be submitted inside φ, there is no guarantee that the clause on the outermost level is used in the derivation. A counterexample is described in detail in [5]. Briefly, the counterexample leads to $\vdash (p \to q) \to \Box_p\ q$, which is not valid. For $\Box_p\ q$ to be true in a policy context w, it requires either q to already be true, or both p and q to be true. On the other hand, $p \to q$ is true if either q is true, or $\neg p$ is true. Note that $(p \to q)$ and $\Box_p\ q$ are not logically equivalent to each other. See the end of Section 3.1 for details.

Axiom (Fun) allows negation and the modal operator to commute with each other: if after submitting the policy γ, φ is not derivable, then it is not the case that submitting the policy γ will make φ derivable, and vice versa.

Axiom (Perm) states that submitting polices one by one is equivalent to submitting them all at once.

The proof rule (Mon) states that if φ can be derived given a set of policies γ', then φ can be derived given a set of logically stronger policies γ. This is only true if φ does not contain negation. Consider an example where $\varphi = \neg p$. Even though $\vdash q \wedge p \to q$, the formula $\Box_q\ \neg p \to \Box_{q,p}\ \neg p$ is not valid.

The proof system defined in Figure 3 can be used to prove that a probing attack is useful for the attacker. For example, a proof of $\vdash \varphi_1 \wedge \varphi_2 \to U.\text{lowGPA(B)}$ can be constructed.

The main property of the proof system is that it is sound and complete with regard to semantics. In addition to the derivability semantics, Becker et al. also defined Kripke semantics, as is common for modal logic. The axiomatized proof system is sound and complete with regard to both models, where $\vDash_{\text{TM}} \varphi$ denotes that φ is valid in the Kripke model.

Theorem 3.1. $\Vdash \varphi \Leftrightarrow \vDash_{TM} \varphi \Leftrightarrow \vdash \varphi$.

4 SEQUENT CALCULUS

Axiomatized systems are inconvenient for automated theorem proving and for studying meta-properties of the logic system. For instance, Becker et al. have to prove a series of complex lemmas to manipulate formula contexts. We develop a Gentzen-style sequent calculus for counterfactual reasoning of Horn clauses. A left and a right rule is defined for each connective. The left rule, read from the bottom up, shows how to deconstruct a connective on the left and the right rule, read from the top down, shows how to construct a connective on the right. Derivation rules for each of the logical connectives are defined independently from each other. The sequent calculus rules provide inductive principles to prove meta-properties of the proof system.

Our formalism is inspired by the formalization of classical modal S5 by Murphy et al. [28]. This formalization is a hybrid logic where elements in the Kripke semantics are present in the sequent calculus rules. We write φ @ w to denote the counterfactual condition that if w were submitted, then φ would have been true. Here, a world, w, corresponds to a set of policies that are true in that world. The relation between worlds are also made explicit in the proof rules.

4.1 Syntax

Formulas φ and policies γ are the same as in the previous section. We define additional constructs for our sequent calculus below.

$$
\begin{array}{lll}
\textit{Judgment} & J & ::= \varphi\ @\ w \\
\textit{Contexts} & \Delta, \Gamma & ::= \cdot \mid \Delta, J \\
\textit{Positive Forms} & \varphi^+ & ::= \gamma^+ \mid \varphi^+ \wedge \varphi'^+ \mid \varphi^+ \vee \varphi'^+ \\
 & & \quad \mid \Box_\gamma\ \varphi^+ \mid \varphi^- \to \varphi^+ \mid \neg\varphi^- \\
\textit{Negative Forms} & \varphi^- & ::= \bot \mid \varphi^- \wedge \varphi'^- \mid \varphi^- \vee \varphi'^- \\
 & & \quad \mid \Box_\gamma\ \varphi^- \mid \neg\varphi^+ \mid \varphi^+ \to \varphi^-
\end{array}
$$

We define positive and negative formulas. Intuitively, positive formulas are the ones that cannot be made false by strengthening the policies and negative formulas are the ones that cannot be made false by weakening the policies. A judgment, denoted J, is a counterfactual condition which states that if w were submitted, then the formula φ is derivable. We write Γ and Δ to denote the contexts that contain a set of counterfactual conditional judgments.

4.2 Derivation Rules

The main judgment of our logic is: $\Gamma \vdash_\Sigma \Delta$. This judgement means that if all of the judgments in Γ are true, then one of the judgments in Δ is true. This follows the sequent calculus for classical logic. The context Σ is a finite set of atomic propositions which includes all propositions relevant to the application. For instance, given a trust

$$\frac{\mathsf{symbols}(\Gamma, p\ @\ w, \Delta) \subseteq \Sigma}{\Gamma, p\ @\ w \vdash_\Sigma p\ @\ w, \Delta}\ \textsc{Init}$$

$$\frac{w \Vdash_d p \qquad \Gamma, p\ @\ w \vdash_\Sigma \Delta}{\Gamma \vdash_\Sigma \Delta}\ \textsc{PolCut}$$

$$\frac{\Gamma, p\ @\ (w, w', q), q\ @\ w, p\ @\ (w, w') \vdash_\Sigma \Delta}{\Gamma, p\ @\ (w, w', q), q\ @\ w \vdash_\Sigma \Delta}\ \textsc{PolL-W-At}$$

$$\frac{\Gamma, p\ @\ (w, w', \Box_{\vec{a}}\, b), b\ @\ (w, \vec{a}), p\ @\ (w, w') \vdash_\Sigma \Delta}{\Gamma, p\ @\ (w, w', \Box_{\vec{a}}\, b), b\ @\ (w, \vec{a}) \vdash_\Sigma \Delta}\ \textsc{PolL-W-}\Box$$

$$\frac{\begin{array}{c}\Gamma, s\ @\ (q \coloneq p_1, \cdots, p_n, w), s\ @\ w \vdash_\Sigma \Delta \\ \Gamma, s\ @\ (q \coloneq p_1, \cdots, p_n, w), s\ @\ (q, w), \\ p_1\ @\ w, \cdots, p_n\ @\ w \vdash_\Sigma \Delta\end{array}}{\Gamma, s\ @\ (q \coloneq p_1, \cdots, p_n, w) \vdash_\Sigma \Delta}\ \textsc{Dlog}$$

$$\frac{\Gamma, p\ @\ w_1, p\ @\ w_2 \vdash_\Sigma \Delta \qquad \wedge(w_2)\ @\ \top \vdash_\Sigma \wedge(w_1)\ @\ \top}{\Gamma, p\ @\ w_1 \vdash_\Sigma \Delta}\ \textsc{Conv}$$

$$\frac{\Gamma \vdash_\Sigma \varphi\ @\ (w, \overline{\gamma}), \Box_\gamma\, \varphi\ @\ w, \Delta}{\Gamma \vdash_\Sigma \Box_\gamma\, \varphi\ @\ w, \Delta}\ \Box\text{R}$$

$$\frac{\Gamma, \Box_\gamma\, \varphi\ @\ w, \varphi\ @\ (w, \overline{\gamma}) \vdash_\Sigma \Delta}{\Gamma, \Box_\gamma\, \varphi\ @\ w \vdash_\Sigma \Delta}\ \Box\text{L}$$

Figure 4: Sequent calculus for policies

management system, Σ would be every atomic proposition in that system, which is finite. We can prove as an invariant of our sequent rules that if $\Gamma \vdash_\Sigma \Delta$, then the atomic propositions in context Γ and Δ are a subset of Σ.

We write $\mathsf{symbols}(J)$ to denote the set of atomic propositions in J. We lift it to the logical context and write $\mathsf{symbols}(\Gamma)$ to denote the set of atomic propositions in Γ.

The sequent rules for the connectives from propositional logic are straightforward. They are shown in Figure 5. We omit Σ when it is clear from the context. We explain rules directly related to counterfactual reasoning (Figure 4).

Rule Init states that if we assume that an atomic predicate p can be derived from policies in w, then we can conclude the same. The side condition $\mathsf{symbols}(\Gamma, p\ @\ w, \Delta) \subseteq \Sigma$ ensures that the proof does not include propositions that are not relevant to the system. PolCut, PolL-W-At, PolL-W-$\Box$, Dlog, and Conv resemble left rules, so they are read from the bottom up.

Rule PolCut connects the derivability of a proposition from a set of Horn clauses to counterfactual reasoning. If p is derivable from a policy w, then it must be the case that the counterfactual condition, "if w were submitted, then p is derivable" is true. Therefore, it can be added to the antecedents. While it is not included as a condition here, it will be enforced later that the propositions in p and w are confined to atomic propositions from the set Σ.

$$\frac{\Gamma, \varphi_1\ @\ w \vdash_\Sigma \varphi_2\ @\ w, \varphi_1 \to \varphi_2\ @\ w, \Delta}{\Gamma \vdash_\Sigma \varphi_1 \to \varphi_2\ @\ w, \Delta}\ \to\text{R}$$

$$\frac{\begin{array}{c}\Gamma, \varphi_1 \to \varphi_2\ @\ w \vdash_\Sigma \varphi_1\ @\ w, \Delta \\ \Gamma, \varphi_2\ @\ w, \varphi_1 \to \varphi_2\ @\ w \vdash_\Sigma \Delta\end{array}}{\Gamma, \varphi_1 \to \varphi_2\ @\ w \vdash_\Sigma \Delta}\ \to\text{L}$$

$$\frac{\Gamma \vdash_\Sigma \varphi_1\ @\ w, \varphi_1 \vee \varphi_2\ @\ w, \Delta}{\Gamma \vdash_\Sigma \varphi_1 \vee \varphi_2\ @\ w, \Delta}\ \vee\text{R1}$$

$$\frac{\Gamma \vdash_\Sigma \varphi_2\ @\ w, \varphi_1 \vee \varphi_2\ @\ w, \Delta}{\Gamma \vdash_\Sigma \varphi_1 \vee \varphi_2\ @\ w, \Delta}\ \vee\text{R2}$$

$$\frac{\begin{array}{c}\Gamma, \varphi_1 \vee \varphi_2\ @\ w, \varphi_1\ @\ w \vdash_\Sigma \Delta \\ \Gamma, \varphi_1 \vee \varphi_2\ @\ w, \varphi_2\ @\ w \vdash_\Sigma \Delta\end{array}}{\Gamma, \varphi_1 \vee \varphi_2\ @\ w \vdash_\Sigma \Delta}\ \vee\text{L}$$

$$\frac{\begin{array}{c}\Gamma \vdash_\Sigma \varphi_1\ @\ w, \varphi_1 \wedge \varphi_2\ @\ w, \Delta \\ \Gamma \vdash_\Sigma \varphi_2\ @\ w, \varphi_1 \wedge \varphi_2\ @\ w, \Delta\end{array}}{\Gamma \vdash_\Sigma \varphi_1 \wedge \varphi_2\ @\ w, \Delta}\ \wedge\text{R}$$

$$\frac{\Gamma, \varphi_1 \wedge \varphi_2\ @\ w, \varphi_1\ @\ w \vdash_\Sigma \Delta}{\Gamma, \varphi_1 \wedge \varphi_2\ @\ w \vdash_\Sigma \Delta}\ \wedge\text{L1}$$

$$\frac{\Gamma, \varphi_1 \wedge \varphi_2\ @\ w, \varphi_2\ @\ w \vdash_\Sigma \Delta}{\Gamma, \varphi_1 \wedge \varphi_2\ @\ w \vdash_\Sigma \Delta}\ \wedge\text{L2}$$

$$\frac{\Gamma, \neg\varphi\ @\ w \vdash_\Sigma \varphi\ @\ w, \Delta}{\Gamma, \neg\varphi\ @\ w \vdash_\Sigma \Delta}\ \neg\text{L} \qquad \frac{\Gamma, \varphi\ @\ w \vdash_\Sigma \neg\varphi\ @\ w, \Delta}{\Gamma \vdash_\Sigma \neg\varphi\ @\ w, \Delta}\ \neg\text{R}$$

$$\frac{\mathsf{symbols}(\Gamma, \top\ @\ w, \Delta) \subseteq \Sigma}{\Gamma \vdash_\Sigma \top\ @\ w, \Delta}\ \top\text{R}$$

$$\frac{\mathsf{symbols}(\Gamma, \bot\ @\ w, \Delta) \subseteq \Sigma}{\Gamma, \bot\ @\ w \vdash_\Sigma \Delta}\ \bot\text{L}$$

Figure 5: Sequent calculus for classical propositional logic

PolL-W-At and PolL-W-$\Box$ are rules for simplifying policies based on the derivability of Horn clauses. PolL-W-At says that if we assume p is true given w, w', and q, *and* we assume that q is true given w, then it is safe to assume that p is true given only w and w'; q is redundant. PolL-W-$\Box$ is similar except that the redundant policy is of form $\Box_{\vec{a}}\, b$.

The Dlog rule introduces counterfactual reasoning of Horn clauses into the sequent rules. It says that if we know that an atomic proposition s is true given the policies in w and $q \coloneq p_1, \cdots, p_n$, then it must be the case that either s is already true given w; or if rule $q \coloneq p_1, \cdots, p_n$ were not present, then s could be derived. The latter part is reflected in the last premise, where the body propositions of the rule are derivable in w, and s is derivable under w and the head of the rule (q).

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\cdots, c\ @\ (w,a,b), a\ @\ w, c\ @\ (w,b) \vdash c\ @\ (w,b) \cdots}\ \textsc{Init}}{\cdots, c\ @\ (w,a,b), a\ @\ w \vdash c\ @\ (w,b), \Box_b\ c\ @\ w, \cdots}\ \textsc{PolL-W-At}}{\cdots, \Box_{a\wedge b}\ c\ @\ w, c\ @\ (w,a,b), a\ @\ w \vdash \Box_b\ c\ @\ w, \cdots}\ \Box R}{\cdots, \Box_{a\wedge b}\ c\ @\ w, a\ @\ w \vdash \Box_b\ c\ @\ w, \neg a\ @\ w}\ \Box L}{\neg\Box_b\ c\ @\ w, \Box_{a\wedge b}\ c\ @\ w \vdash \Box_b\ c\ @\ w, \neg a\ @\ w}\ \neg R}{\neg\Box_b\ c\ @\ w, \Box_{a\wedge b}\ c\ @\ w \vdash \neg a\ @\ w}\ \neg L$$

Figure 6: Simple probing attack proof

The next rule allows us to change the world (policies) under which a formula is derived. Conv makes explicit the relation between worlds. If we assume that p is derivable after submitting polices w_1 ($p\ @\ w_1$), then it is safe to assume $p\ @\ w_2$ given that w_1 is weaker than w_2. More simply, this rule states that if p is derivable given w_1 and w_1 is derivable from w_2, then p should also be derivable from w_2. Similar to the PolCut rule, the newly introduced world w_2 can only be composed of symbols from Σ.

The left and right rules for the $\Box$ connective simply add the policy γ from the index of the modal operator $\Box$ to the world under which the formula is queried. The right rule states that if φ is true given w and γ, then $\Box_\gamma\ \varphi$ is true given only w. The left rule says that to use $\Box_\gamma\ \varphi\ @\ w$ in the context, decompose it and use the fact that φ is true under w and $\overline{\gamma}$. This rule is closely related to the Kripke semantics of the $\Box$ modality. The Kripke semantics require φ to be true in all worlds that are reachable from w with distance γ. In this rule, we only select one such world, which is $(w, \overline{\gamma})$. As a result, we compensate with our conversion rule (see Section 6).

4.3 Example Derivations

We show in Figure 6 the proof for the counterfactual reasoning: $\vdash \neg\Box_b\ c \wedge \Box_{a\wedge b}\ c \rightarrow \neg a$. The probes tell us the following. If we submit b, c cannot be derived. However, if we submit both a and b, c can be derived. From the probes, we can conclude that in the current system, a cannot be true; otherwise, submitting b would have allowed the derivation of c. Reading from the bottom up, first ¬L, ¬R, □L, and □R rules are used to reach the subgoal: if c is derivable given w, a, and b, and a is derivable from w, then c is derivable from w and b. Intuitively, this is correct because w subsumes a. This subgoal can be discharged by applying the PolL-W-AT rule. We complete the proof by applying the Init rule.

Next, we show the proof of the probing attack discussed in Section 1, formalized in Section 3.1. We use the following abbreviation: $A_a = U.\mathsf{canAccessGym}(A)$, $G_b = U.\mathsf{lowGPA}(B)$, and $R_a = A.\mathsf{registerForGym}$. We write $\mathcal{E}$ :: to label derivations so they can be referred to later.

$$\dfrac{\dfrac{\mathcal{E}_1 :: \neg A_a\ @\ \top, \cdots, A_a\ @\ \top \vdash G_b\ @\ \top \quad \mathcal{E}_2 :: \cdots, G_b\ @\ \top \vdash G_b\ @\ \top}{\neg A_a\ @\ \top, \ldots, A_a\ @\ (\top, R_a \text{ :- } G_b) \vdash G_b\ @\ \top}\ \textsc{DLog}}{\neg A_a\ @\ \top, \Box_{R_a \text{ :- } G_b}\ A_a\ @\ \top \vdash G_b\ @\ \top}\ \Box L$$

Once we apply the DLog rule, we only need to prove that if A currently has access to the gym then B has a low GPA ($\mathcal{E}_1$) and that if B has a low GPA, then B has a low GPA ($\mathcal{E}_2$). $\mathcal{E}_1$ contains a contradiction on the left, because our first probe says that currently A does not have access to the gym. We can apply ¬L rule to discharge it. $\mathcal{E}_2$ can be proven by applying the Init rule.

The last example proves a more complex probing attack from [5]. Consider the following formula:

$$s\ @\ a, \neg s\ @\ (a \text{ :- } b), s\ @\ (a \text{ :- } b, b \text{ :- } x) \vdash x\ @\ \top$$

x is a sensitive credential that should not be leaked to the attacker. On the left of the turnstile are three probes by the attacker. The proof shows that x is detectable under these probes. Informally, from the first probe, the attacker knows that if a is submitted, then s is true. The second probe shows that if $(a \text{ :- } b)$ is submitted, then s cannot be derived. From these two probes, the attacker knows that b cannot be true. Otherwise, the second probe would have shown that s is derivable after $(a \text{ :- } b)$ is submitted. The third probe shows that s is derivable if two policies:$(a \text{ :- } b$ and $b \text{ :- } x)$ are submitted. We already know from the second probe that $(a \text{ :- } b)$ alone cannot make s derivable, so $(b \text{ :- } x)$ is crucial in deriving s. Since we already know that b is not true in the current context, it must be the case that the body of $(b \text{ :- } x)$ is true, which is the secret that the attacker tries to guess. We explain the proofs using our sequent calculus rules, reading from the bottom up (Figure 7). We mark formulas used on the rules in red.

To avoid using Cut, our proof analyzes the last probe first. We apply DLog rule on the last probe and deconstruct the rule $(b \text{ :- } x)$. The first subgoal $\mathcal{D}$ contains a contradiction. This basically states that there is no way s can be derived only from $(a \text{ :- } b)$. The second subgoal $\mathcal{E}$ analyzes the situation where x is true under the policy $(a \text{ :- } b)$ and s is derivable assuming b and $(a \text{ :- } b)$ are derivable. To prove $\mathcal{E}$, we apply DLog rule and analyze the assumption: x is true under the policy $(a \text{ :- } b)$. There are two cases: either x is true without using $a \text{ :- } b$, in which case we succeed in proving that $x\ @\ \top$; or x is true given a, and b is true. The second case is denoted $\mathcal{E}'$. The proof of $\mathcal{E}'$ essentially derives a contradiction: s can be derived from $a \text{ :- } b$. This is by using the ¬L first, then applying the PolL-W-At rule which shows that s must be derivable from $a \text{ :- } b$, because b is true and s can be derived from b and $a \text{ :- } b$.

In the end, we cover all the cases, and show that x is detectable under the three probes.

5 METATHEORY

We prove that our sequent calculus has cut elimination, is consistent, and is sound and complete with regard to the axiomatized

$$\dfrac{\begin{array}{c}\mathcal{D} :: s \mathbin{@} a, \neg s \mathbin{@} (a \coloneq b), s \mathbin{@} (a \coloneq b, b \coloneq x), \underline{s \mathbin{@} (a \coloneq b)} \vdash x \mathbin{@} \top \\ \mathcal{E} :: s \mathbin{@} a, \neg s \mathbin{@} (a \coloneq b), s \mathbin{@} (a \coloneq b, b \coloneq x), s \mathbin{@} (b, a \coloneq b), x \mathbin{@} (a \coloneq b) \vdash x \mathbin{@} \top\end{array}}{s \mathbin{@} a, \neg s \mathbin{@} a \coloneq b, s \mathbin{@} (a \coloneq b, b \coloneq x) \vdash x \mathbin{@} \top}\ \textsc{DLog}$$

$\mathcal{D}$ can be proving using ¬L and Init

$$\mathcal{E} = \dfrac{\cdots, \underline{x \mathbin{@} \top} \vdash \underline{x \mathbin{@} \top} \qquad \mathcal{E}' :: \cdots, \neg s \mathbin{@} (a \coloneq b), s \mathbin{@} (b, a \coloneq b), x \mathbin{@} (a, \top), b \mathbin{@} \top \vdash x \mathbin{@} \top}{s \mathbin{@} a, \neg s \mathbin{@} (a \coloneq b), s \mathbin{@} (b, a \coloneq b), x \mathbin{@} a \coloneq b \vdash x \mathbin{@} \top}\ \textsc{DLog}$$

$$\mathcal{E}' = \dfrac{\dfrac{\dfrac{}{\cdots, s \mathbin{@} (b, a \coloneq b), s \mathbin{@} (a \coloneq b), b \mathbin{@} \top \vdash s \mathbin{@} (a \coloneq b), x \mathbin{@} \top}\ \textsc{Init}}{\cdots, s \mathbin{@} (b, a \coloneq b), b \mathbin{@} \top \vdash s \mathbin{@} (a \coloneq b), x \mathbin{@} \top}\ \textsc{PolL-W-At}}{\cdots, \neg s \mathbin{@} (a \coloneq b), s \mathbin{@} (b, a \coloneq b), b \mathbin{@} \top \vdash x \mathbin{@} \top}\ \neg L$$

Figure 7: Complex probing attack proof

formulation in Section 3. At the end of this section, we prove correct a defense against probing attacks. The proof inducts over the structure of the derivation, which is a huge benefit of sequent calculus compared to axiomatized systems.

5.1 Admissibility of Cut (Cut Elimination)

One of the most important properties of a sequent calculus is the admissibility of cut (Theorem 4). We write $\mathcal{D} ::$, $\mathcal{E} ::$, or $\mathcal{F} ::$ to label derivations so they can be referred to later. We can prove the admissibility of cut using similar techniques outlined by Pfenning [33].

Theorem 4 (Admissibility of Cut)**.**
If $\mathcal{D} :: \Gamma \vdash \varphi \mathbin{@} w, \Delta$ *and* $\mathcal{E} :: \Gamma, \varphi \mathbin{@} w \vdash \Delta$*, then* $\exists \mathcal{F} :: \Gamma \vdash \Delta$

Proof (sketch): Following Pfenning [33], we refer to a formula that is introduced on the left or right as a principal formula. For the Init rule, the principal formula is the formula appearing on both sides. For the ⊤R and ⊥L rules, the principal formula is the $\top \mathbin{@} w$ formula appearing on the right and the $\bot \mathbin{@} w$ formula appearing on the left, respectively. We refer to all other formulas as side formulas.

The proof is by nested induction, first over the structure of the cut formula, φ, and then over the structures of $\mathcal{D}$ and $\mathcal{E}$. All proof cases fall into 4 categories (many of these cases overlap), more details may be found in Appendix C.

Base cases: Either $\mathcal{D}$ or $\mathcal{E}$ ends in Init, ⊤ R, or ⊥ L

- Either $\mathcal{D}$ or $\mathcal{E}$ ends in Init and the other derivation is arbitrary. The cut formula is $p \mathbin{@} w$. Applying the contraction lemma on the arbitrary derivation will result in $\mathcal{F}$.
- $\mathcal{D}$ is an arbitrary derivation and $\mathcal{E}$ ends in ⊤R. Then, the cut formula is an arbitrary $\varphi \mathbin{@} w$. This case is covered by side cases on $\mathcal{E}$.
- $\mathcal{D}$ ends in ⊤R and $\mathcal{E}$ is an arbitrary derivation. Then, the cut formula is $\top \mathbin{@} w$. This case is covered by side cases on $\mathcal{E}$.
- $\mathcal{D}$ ends in ⊥L and $\mathcal{E}$ is an arbitrary derivation. Then, the cut formula is an arbitrary $\varphi \mathbin{@} w$. This case is covered by side cases on $\mathcal{D}$.
- $\mathcal{D}$ is an arbitrary derivation and $\mathcal{E}$ ends in ⊥L. Then, the cut formula is $\bot \mathbin{@} w$. This case is covered by side cases on $\mathcal{D}$.

Principal cases: The cut formula is the principal formula in both $\mathcal{D}$ and $\mathcal{E}$. $\mathcal{D}$ and $\mathcal{E}$ end in matching left and right rules of the same connective. (1) Apply the induction hypothesis on the cut formula, $\mathcal{D}$, and each of the subderivations of $\mathcal{E}$. (2) Apply the induction hypothesis on the cut formula, each of the subderivations of $\mathcal{D}$, and $\mathcal{E}$. Applying the induction hypothesis on a subformula of the cut formula and the results of (1) and (2) gives $\mathcal{F}$.

Special cases: $\mathcal{E}$ ends in PolCut, PolL-W-At, PolL-W-□, DLog, or Conv and the cut formula is one of the formulas used by the rule. These cuts may be of interest since they involve our new rules, but they do not actually require separate proof cases. All of these types of cuts are covered by other cases, outlined below.

- $\mathcal{D}$ ends in Init. This case is covered by the base cases where $\mathcal{D}$ is Init and $\mathcal{E}$ is an arbitrary derivation.
- The cut formula is a side formula in $\mathcal{D}$. This case is covered by the side cases on $\mathcal{D}$.

Side cases on $\mathcal{D}$: The cut formula is a side formula in $\mathcal{D}$. There is a case for $\mathcal{D}$ ending in every rule. $\mathcal{E}$ is always an arbitrary derivation.

- $\mathcal{D}$ ends in Init, ⊤ R, or ⊥ L: We can directly apply the rule which $\mathcal{D}$ ends in to obtain $\mathcal{F}$.
- Otherwise: We can apply the rule which $\mathcal{D}$ ends in on the result of applying the induction hypothesis to the cut formula, the subderivations of $\mathcal{D}$, and $\mathcal{E}$, to obtain $\mathcal{F}$.

Side cases on $\mathcal{E}$: The cut formula is a side formula in $\mathcal{E}$. There is a case for $\mathcal{E}$ ending in every rule. $\mathcal{D}$ is always an arbitrary derivation.

- $\mathcal{E}$ ends in Init, ⊤ R, or ⊥ L: We can directly apply the rule which $\mathcal{E}$ ends in to obtain $\mathcal{F}$.
- Otherwise: We can apply the rule which $\mathcal{E}$ ends in on the result of applying the induction hypothesis to the cut formula, $\mathcal{D}$, and the subderivations of $\mathcal{E}$, to obtain $\mathcal{F}$.

□

5.2 Consistency

Normally, with the admissibility of cut, the consistency of the logic becomes a trivial proof by showing that there is no rule to derive

$\cdot \vdash \bot @ w$. However, for us, the PolCut rule could be used. To prove consistency, we need to generalize the statement and induct over the size of the derivation. Theorem 5 allows any counterfactual condition of form $p @ w$ in Γ. The ordinary consistency property follows directly from it.

Theorem 5 (Consistency)**.**
$\forall n, \nexists \mathcal{E}, \nexists w, \nexists \Gamma$ *such that* $\mathcal{E} :: \Gamma \vdash \bot @ w$ *and* $|\mathcal{E}| \leq n$ *where* $\forall J \in \Gamma$, $\mathcal{J} = p @ w$.

5.3 Soundness

For soundness, we need to show that if φ is derivable using our sequent rules, it is also derivable using the proof system in Figure 3. Instead of proving this directly, we prove the soundness of our sequent calculus with regard to the formula semantics (Theorem 6). We write $\bigwedge \Gamma$ to denote the conjunction of the counterfactuals that correspond to judgments in Γ. That is, the counterfactual that corresponds to the judgment $\varphi @ w$ is $\Box_w \varphi$. Similarly, $\bigvee \Delta$ is the disjunction of the counterfactuals that corresponds to judgments in Δ. In combination with Theorem 3.1, Theorem 6 allows us to connect formulas derivable in our sequent rules to those derivable using the axioms.

Theorem 6 (Soundness)**.**
If $\mathcal{E} :: \Gamma \vdash \Delta$, *then* $\forall w, w \Vdash \neg(\bigwedge \Gamma) \vee \bigvee \Delta$.

The proof is by induction on the structure of $\mathcal{E}$. Most cases are straightforward. The most interesting case is where $\mathcal{E}$ ends in Conv. The I.H. on the first premise gives us

$$\mathcal{E}_1' :: w \Vdash \neg \bigvee \Gamma \vee \Box_{w_1} \neg p \vee \Box_{w_2} \neg p \vee \bigvee \Delta$$

but we want to show

$$w \Vdash \neg \bigvee \Gamma \vee \Box_{w_1} \neg p \vee \bigvee \Delta$$

There are several cases to consider here; namely, when $w \Vdash \neg \bigvee \Gamma$, when $w \Vdash \Box_{w_1} \neg p$, and so on. The important case here is when it's not immediately obvious if we can remove $\Box_{w_2} \neg p$ from $\mathcal{E}_1'$, the case where

$$w \Vdash \Box_{w_2} \neg p$$

We need to conclude that $w, w_2 \Vdash \neg p$ implies $w, w_1 \Vdash \neg p$. This follows from our intuition about Conv because the last premise of Conv, $\wedge(w_2) @ \top \vdash \wedge(w_1) @ \top$, suggests that w_2 contains stronger policies that w_1. If p is not derivable from w_2, then it should not be derivable from w_1. However, it turns out that the I.H. on the last premise only gives us $\forall w', w' \Vdash \neg(\wedge w_2) \vee \wedge w_1$, which is not strong enough. Instead, we need $w_2 \Vdash \wedge(w_1)$.

In fact, we can prove the following lemma, which states that if a policy γ_2 can be derived from another policy γ_1 using our sequent rules, then γ_2 is stronger than γ_1 under datalog semantics.

Lemma 7. $\forall \gamma_1, \gamma_2$, *If* $\mathcal{E} :: \gamma_1 @ \top \vdash \gamma_2 @ \top$ *then* $\overline{\gamma_1} \Vdash \gamma_2$.

This lemma confirms that our sequent rules $\vdash$ reason about counterfactual conditions. If, by using our main judgment, we can prove that γ_2 can be derived with only the additional assumption that γ_1 can be derived with no additional policy assumptions, then γ_2 must be derivable from γ_1.

Proving Lemma 7 is non-trivial. Inducting over the structure of $\mathcal{E} :: \gamma_1 @ \top \vdash_\Sigma \gamma_2 @ \top$ proves insufficient because PolCut and Conv can introduce judgements that our induction hypothesis does not account for. We want to show that we can safely disregard these judgements because they are derivable facts and are made redundant by what we already know. Intuitively, given a sub-derivation of $\gamma_1 @ \top \vdash_\Sigma \gamma_2 @ \top$ of the form $\Gamma, \Gamma_A \vdash_\Sigma \Delta$, where Γ_A contains all the new judgments introduced via PolCut and Conv, it is the case that $\bigwedge \Gamma \Vdash \bigwedge \Gamma_A$. If $p @ w$ is introduced by PolCut, we already know $w \Vdash_d p$. If $p @ w_2$ is introduced by Conv, we already have $p @ w_1$ in Γ, and w_2 is stronger than w_1 (by I.H.), so $p @ w_2$ can be derived from Γ.

To formally prove this statement, we first define a new calculus containing labels. The labels allow us to track which judgements were introduced by PolCut and Conv, as well as track which judgements were derived from those judgements. This will allow us to distinguish judgements of the form $\gamma @ \top$ from those of the form $p @ w$. We assign the label 'A' to judgements that have been added by, or are derived from judgements which have been added by, PolCut$^+$ or Conv$^+$. All other judgements are labeled with 'O'.

For example, consider the following rule from our labeled sequent calculus:

$$\frac{\text{symbols}(w_2) \subseteq \Sigma \qquad \Gamma^+, (p @ w_1)_x, (p @ w_2)_A \vdash^+_\Sigma \Delta \qquad (\wedge(w_2) @ \top)_o \vdash^+_\Sigma \wedge(w_1) @ \top}{\Gamma^+, (p @ w_1)_x \vdash^+_\Sigma \Delta}\ \text{Conv}^+$$

$p @ w_2$ is a judgement that is being added by Conv, so it is labeled with an 'A' to indicate this. We know that $\wedge(w_2)$ is derivable after submitting $\top$, so it can safely be labeled with an 'O'. The remaining rules, as well as details about the syntax and notation, may be found in Appendix A.

Next, we prove that derivations using the original sequent rules can be annotated, resulting in derivations in the labeled sequent rules. Finally, we prove a more general lemma, which formalizes our intuition of discarding judgments introduced via PolCut and Conv, discussed earlier. We show a less precise and simplified version of the lemma below to illustrate how the lemma relates derivations in the marked up sequent rules to formula semantics.

Lemma 8. *If* $\Gamma^+, \Gamma_A^+ \vdash^+_\Sigma \Delta$ *and* $\bigwedge \Gamma^+ \Vdash \bigwedge \Gamma_A^+$ *and judgments in the contexts are can only be form* $\gamma @ \top$, $q @ w$ *then* $\exists \varphi @ w \in \Delta$ *s.t.* $\bigwedge \Gamma^+ \Vdash \Box_w \varphi$.

In the above lemma Γ_A^+ contains all the judgments that are labeled 'A', which means that it is derived from judgments injected by PolCut and Conv rules. The lemma states that if judgments marked by the label A are derivable by counterfactuals in the rest of the context, then one of the counterfactuals in Δ can be derived from the conjunction of counterfactuals in Γ^+, which contain only judgments decomposed from the judgments in the final conclusion. Lemma 7 is a corollary. The precise lemmas are defined and proven in Appendix A.

5.4 Completeness

For completeness, we prove that for every provable formula φ in the axiomatized system (Figure 3), we can derive $\varphi @ w$ for any w (Theorem 9).

Theorem 9 (Completeness)**.** *If* $\vdash \varphi$, *then* $\forall w, \cdot \vdash \varphi @ w$.

Axiom Cl1

$$\dfrac{\dfrac{\dfrac{}{\varphi_1 \mathbin{@} w, \varphi_2 \mathbin{@} w \vdash_\Sigma \varphi_1 \mathbin{@} w, \cdots}\textsc{Init}}{\varphi_1 \mathbin{@} w \vdash_\Sigma \varphi_2 \to \varphi_1 \mathbin{@} w, \cdots}\to R}{\vdash_\Sigma \varphi_1 \to (\varphi_2 \to \varphi_1) \mathbin{@} w}\to R$$

Axiom Cl2

$$\dfrac{\dfrac{\dfrac{\mathcal{E} :: \varphi_1 \to (\varphi_2 \to \varphi_3) \mathbin{@} w, \varphi_1 \to \varphi_2 \mathbin{@} w, \varphi_1 \mathbin{@} w \vdash_\Sigma \varphi_3 \mathbin{@} w, \cdots}{\varphi_1 \to (\varphi_2 \to \varphi_3) \mathbin{@} w, \varphi_1 \to \varphi_2 \mathbin{@} w \vdash_\Sigma \varphi_1 \to \varphi_3 \mathbin{@} w, \cdots}\to R}{\varphi_1 \to (\varphi_2 \to \varphi_3) \mathbin{@} w \vdash_\Sigma ((\varphi_1 \to \varphi_2) \to (\varphi_1 \to \varphi_3) \mathbin{@} w, \cdots}\to R}{\vdash_\Sigma (\varphi_1 \to \varphi_2 \to \varphi_3) \to ((\varphi_1 \to \varphi_2) \to (\varphi_1 \to \varphi_3)) \mathbin{@} w}\to R$$

We complete this derivation by repeatedly applying →L and Init to $\mathcal{E}$.

Axiom Cl3

$$\dfrac{\dfrac{\mathcal{E} :: \neg\varphi_1 \to \neg\varphi_2 \mathbin{@} w \vdash_\Sigma \neg\varphi_1 \mathbin{@} w, \varphi_2 \to \varphi_1 \mathbin{@} w, \cdots \quad \mathcal{D} :: \cdots, \neg\varphi_2 \mathbin{@} w \vdash_\Sigma \varphi_2 \to \varphi_1 \mathbin{@} w, \cdots}{\neg\varphi_1 \to \neg\varphi_2 \mathbin{@} w \vdash_\Sigma \varphi_2 \to \varphi_1 \mathbin{@} w, \cdots}\to L}{\vdash_\Sigma (\neg\varphi_1 \to \neg\varphi_2) \to (\varphi_2 \to \varphi_1) \mathbin{@} w}\to R$$

$$\mathcal{E} = \dfrac{\dfrac{\dfrac{\dfrac{}{\cdots, \varphi_2 \mathbin{@} w, \varphi_1 \mathbin{@} w \vdash_\Sigma \varphi_1 \mathbin{@} w, \cdots}\textsc{Init}}{\cdots, \varphi_2 \mathbin{@} w \vdash_\Sigma \varphi_1 \mathbin{@} w, \neg\varphi_1 \mathbin{@} w, \cdots}\neg R}{\neg\varphi_1 \to \neg\varphi_2 \mathbin{@} w \vdash_\Sigma \neg\varphi_1 \mathbin{@} w, \varphi_2 \to \varphi_1 \mathbin{@} w, \cdots}\to R$$

$$\mathcal{D} = \dfrac{\dfrac{\dfrac{}{\cdots, \varphi_2 \mathbin{@} w \vdash_\Sigma \varphi_2 \mathbin{@} w, \varphi_1 \mathbin{@} w, \cdots,}\textsc{Init}}{\cdots, \neg\varphi_2 \mathbin{@} w, \varphi_2 \mathbin{@} w \vdash_\Sigma \varphi_1 \mathbin{@} w, \cdots,}\neg L}{\cdots, \neg\varphi_2 \mathbin{@} w \vdash_\Sigma \varphi_2 \to \varphi_1 \mathbin{@} w, \cdots}\to R$$

Axiom Fun

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\cdots, \varphi \mathbin{@} (w, \overline{\gamma}) \vdash_\Sigma \varphi \mathbin{@} (w, \overline{\gamma}), \cdots}\textsc{Init}}{\Box_\gamma \varphi \mathbin{@} w \vdash_\Sigma \varphi \mathbin{@} (w, \overline{\gamma}), \cdots}\Box L}{\cdots \vdash_\Sigma \varphi \mathbin{@} (w, \overline{\gamma}), \neg\Box_\gamma \varphi \mathbin{@} w}\neg R}{\cdots, \neg\varphi \mathbin{@} (w, \overline{\gamma}) \vdash_\Sigma \neg\Box_\gamma \varphi \mathbin{@} w}\neg L}{\Box_\gamma \neg\varphi \mathbin{@} w \vdash_\Sigma \neg\Box_\gamma \varphi \mathbin{@} w}\Box L$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\cdots, \varphi \mathbin{@} (w, \overline{\gamma}) \vdash_\Sigma \varphi \mathbin{@} (w, \overline{\gamma}), \cdots\cdots}\textsc{Init}}{\cdots, \varphi \mathbin{@} (w, \overline{\gamma}) \vdash_\Sigma \Box_\gamma \varphi \mathbin{@} w, \cdots}\Box R}{\cdots, \varphi \mathbin{@} (w, \overline{\gamma}) \vdash_\Sigma \Box_\gamma \varphi \mathbin{@} w, \cdots}\neg L}{\neg\Box_\gamma \varphi \mathbin{@} w, \varphi \mathbin{@} (w, \overline{\gamma}) \vdash_\Sigma \cdots}\neg R}{\neg\Box_\gamma \varphi \mathbin{@} w \vdash_\Sigma \neg\varphi \mathbin{@} (w, \overline{\gamma}), \cdots}\Box R}{\neg\Box_\gamma \varphi \mathbin{@} w \vdash_\Sigma \Box_\gamma \neg\varphi \mathbin{@} w}\Box R$$

Axiom Perm

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\cdots, \varphi \mathbin{@} (w, \overline{\gamma \wedge \gamma'}) \vdash_\Sigma \varphi \mathbin{@} (w, \overline{\gamma}, \overline{\gamma'}), \cdots}\textsc{Init}}{\cdots, \varphi \mathbin{@} (w, \overline{\gamma \wedge \gamma'}) \vdash_\Sigma \Box_{\gamma'} \varphi \mathbin{@} (w, \overline{\gamma}), \cdots}\Box R}{\cdots, \varphi \mathbin{@} (w, \overline{\gamma \wedge \gamma'}) \vdash_\Sigma \Box_\gamma \Box_{\gamma'} \varphi \mathbin{@} w}\Box R}{\Box_{\gamma\wedge\gamma'} \varphi \mathbin{@} w \vdash_\Sigma \Box_\gamma \Box_{\gamma'} \varphi \mathbin{@} w}\Box L$$

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\cdots, \varphi \mathbin{@} (w, \overline{\gamma}, \overline{\gamma'}) \vdash_\Sigma \varphi \mathbin{@} (w, \overline{\gamma \wedge \gamma'}), \cdots}\textsc{Init}}{\cdots, \Box_{\gamma'} \varphi \mathbin{@} (w, \overline{\gamma}) \vdash_\Sigma \varphi \mathbin{@} (w, \overline{\gamma \wedge \gamma'}), \cdots}\Box L}{\Box_\gamma \Box_{\gamma'} \varphi \mathbin{@} w \vdash_\Sigma \varphi \mathbin{@} (w, \overline{\gamma \wedge \gamma'}), \cdots}\Box L}{\Box_\gamma \Box_{\gamma'} \varphi \mathbin{@} w \vdash_\Sigma \Box_{\gamma\wedge\gamma'} \varphi \mathbin{@} w}\Box R$$

Axiom K

$$\dfrac{\dfrac{\mathcal{E} :: \cdots, \varphi_1 \to \varphi_2 \mathbin{@} (w, \overline{\gamma}) \vdash_\Sigma \varphi_1 \mathbin{@} (w, \overline{\gamma}), \Box_\gamma \varphi_1 \to \Box_\gamma \varphi_2 \mathbin{@} w, \cdots \quad \mathcal{D} :: \cdots, \varphi_2 \mathbin{@} (w, \overline{\gamma}) \vdash_\Sigma \Box_\gamma \varphi_1 \to \Box_\gamma \varphi_2 \mathbin{@} w, \cdots}{\cdots, \varphi_1 \to \varphi_2 \mathbin{@} (w, \overline{\gamma}) \vdash_\Sigma \Box_\gamma \varphi_1 \to \Box_\gamma \varphi_2 \mathbin{@} w, \cdots}\to L}{\Box_\gamma (\varphi_1 \to \varphi_2) \mathbin{@} w \vdash_\Sigma \Box_\gamma \varphi_1 \to \Box_\gamma \varphi_2 \mathbin{@} w, \cdots}\Box L}{\vdash_\Sigma \Box_\gamma (\varphi_1 \to \varphi_2) \to \Box_\gamma \varphi_1 \to \Box_\gamma \varphi_2 \mathbin{@} w}\to R$$

$$\mathcal{E} = \dfrac{\dfrac{\dfrac{}{\cdots, \varphi_1 \mathbin{@} (w, \overline{\gamma}) \vdash_\Sigma \varphi_1 \mathbin{@} (w, \overline{\gamma}), \Box_\gamma \varphi_2 \mathbin{@} w, \cdots}\textsc{Init}}{\cdots, \Box_\gamma \varphi_1 \mathbin{@} w \vdash_\Sigma \varphi_1 \mathbin{@} (w, \overline{\gamma}), \Box_\gamma \varphi_2 \mathbin{@} w, \cdots}\Box L}{\cdots, \varphi_1 \to \varphi_2 \mathbin{@} (w, \overline{\gamma}) \vdash_\Sigma \varphi_1 \mathbin{@} (w, \overline{\gamma}), \Box_\gamma \varphi_1 \to \Box_\gamma \varphi_2 \mathbin{@} w, \cdots}\to R$$

$$\mathcal{D} = \dfrac{\dfrac{\dfrac{}{\cdots, \Box_\gamma \varphi_1 \mathbin{@} w, \varphi_2 \mathbin{@} (w, \overline{\gamma}) \vdash_\Sigma \varphi_2 \mathbin{@} (w, \overline{\gamma}), \cdots,}\textsc{Init}}{\cdots, \Box_\gamma \varphi_1 \mathbin{@} w, \varphi_2 \mathbin{@} (w, \overline{\gamma}) \vdash_\Sigma \Box_\gamma \varphi_2 \mathbin{@} w, \cdots}\Box R}{\cdots, \varphi_2 \mathbin{@} (w, \overline{\gamma}) \vdash_\Sigma \Box_\gamma \varphi_1 \to \Box_\gamma \varphi_2 \mathbin{@} w, \cdots}\to R$$

Figure 8: Derivations for the proof of Axioms Cl1, Cl2, Cl3, Fun, Perm, and K

The proof of the completeness theorem shows that all of the axioms can be derived using the sequent calculus rules and that the proof rules are admissible in our sequent calculus. Axioms Cl1, Cl2, Cl3, Fun, Perm, and K can be directly constructed using our sequent rules. The derivations may be found in Figure 8. The remaining axioms can be shown by induction over the structure of φ and we list the lemmas for these axioms below.

Lemma 10 (Axiom C1). $\forall\ \gamma, w, \vdash_\Sigma \gamma\ @\ (\overline{\gamma}, w)$.

Lemma 11 (Axiom C2). $\forall\ \varphi, \gamma, w, \vdash_\Sigma (\Box_\gamma\ \varphi \rightarrow \gamma \rightarrow \varphi)\ @\ w$.

Lemma 12 (Axiom Dlog). $\forall w, \varphi, p, \vec{q}, \varphi,$ *if* φ *is* $\Box$ *free then* $\vdash_\Sigma \Box_{(q\ :\text{-}\ \vec{p})}\ \varphi \rightarrow (\vec{p} \rightarrow q) \rightarrow \varphi\ @\ w$

Lemma 13 (Axiom MP). *If* $\vdash \varphi_1\ @\ w$ *and* $\vdash \varphi_1 \rightarrow \varphi_2\ @\ w$ *then* $\vdash \varphi_2\ @\ w$.

Lemma 14 (Axiom N).
$\forall\ \varphi, w, \gamma,$ *if* $\vdash_\Sigma \varphi\ @\ w,$ *then* $\vdash_\Sigma \Box_\gamma\ \varphi\ @\ w$

The proof of Axiom Mon follows from a more general lemma for positive and negative formulas and requires additional lemmas to remove unnecessary judgements from the context. The proofs of these lemmas and Axiom Mon can be found in Appendix E.

Lemma 15 (Axiom Mon). *If* $\vdash_\Sigma \gamma_1 \rightarrow \gamma_2\ @\ \top$ *then* $\forall w \vdash_\Sigma \Box_{\gamma_2}\ \varphi^+ \rightarrow \Box_{\gamma_1}\ \varphi^+\ @\ w$.

Axiom Mon requires ¬-free formulas. Our Lemma 15 is a more general form that allows not only ¬-free formulas, but also other positive formulas.

6 DISCUSSION

Alternative Formalizations In our initial attempt, we tried to use only one policy context and allow Γ and Δ to only contain formulas. The judgment is of the form: $w; \Gamma \vdash \Delta$. The reading of this judgment is that if policies in w are submitted, and all of the formulas in Γ are derivable, then one of the formulas in Δ is derivable. The precise meaning of the judgment with regard to the model is the following: If $w; \Gamma \vdash \Delta$ then $\forall w', w', w \Vdash \bigwedge \Gamma \rightarrow \bigvee \Delta$.

The □R and □L rules are as follows.

$$\frac{w, \gamma; \Gamma_a, \Gamma \vdash \Delta_b, \Delta \qquad \Gamma_a \text{ contains only positive formulas} \quad \Delta_b \text{ contains only negative formulas}}{w; \Gamma_a, \Box_\gamma\ \Gamma \vdash \Delta_b, \Box_\gamma\ \Delta}\ \Box\text{R'}$$

$$\frac{w; \Box_\gamma\ \varphi, \Gamma \vdash \gamma, \Delta \qquad w; \Box_\gamma\ \varphi, \varphi, \Gamma \vdash \Delta}{w; \Box_\gamma\ \varphi, \Gamma \vdash \Delta}\ \Box\text{L'}$$

We write $\Box_\gamma\ \Gamma$ to denote the formula context resulting from wrapping every formula in Γ with $\Box_\gamma$. The □R' introduces counterfactuals on both sides of the turnstile, except for formulas in contexts Γ_a and Δ_b. The first premise of this rule states that if policies w and γ were submitted, and all the formulas in Γ_a and Γ are derivable, then one of the formulas in Δ_b and Δ is derivable. The policy context in the conclusion contains only w. However, formulas in Γ and Δ are wrapped under $\Box_\gamma$. Because submitting w and γ is the same as submitting w first, then submitting γ, if all of the formulas in $\Box_\gamma\ \Gamma$ are true if w is submitted, it implies that all of the formulas in Γ are true if w and γ is true. The same reasoning applies to Δ. Recall that positive formulas cannot be falsified by strengthening policies and negative formulas cannot be falsified by weakening policies. Formulas in Γ_a are positive, so if they are true under w, then they are true under w and γ. Formulas in Δ_b are negative, so if one of them is true under w and γ, then it is true under w as well. This is why the □R rule is sound. The left rule removes the conditional γ if the policy γ is true already.

This formulation can be shown to be sound and complete with regard to the axiomatized system. However, cut elimination cannot be proven. This is similar to the problem of Prawitz's formulations of necessity [35]. One way to solve this problem as pointed out by Pfenning et al. is to have separate judgments for truths and validity [34]. This is precisely what we do here. We have a separate judgment for counterfactual conditions. Instead of having a shared policy world, we index each formula with its policy world using judgment: $\varphi\ @\ w$.

World Relations and Kripke Semantics Our proof rules explicitly index formulas with policy worlds and operate on policy worlds. There is a strong connection between the Kripke Semantics defined by Becker et al. [5] and how our proof rules manipulate policy worlds.

A Kripke model M is a triple $\langle W, R, V\rangle$, where W is a set of worlds, $R \subseteq \wp(W) \times W \times W$, and $V : \mathbf{At} \rightarrow \wp(W)$. We write $\wp(W)$ to denote the power set of W. The semantics of counterfactuals given the model is as follows:
$w \Vdash_M \Box_\gamma\ \varphi$ iff $\forall w', R_{|\gamma|_M}(w, w') \Rightarrow w' \Vdash_M \varphi$ where $|\gamma|_M = \{w \in W \mid w \Vdash_M \gamma\}$.

Here, the accessibility relation between worlds are indexed by a set of worlds where a policy γ is true ($R_{|\gamma|_M}(w, w')$).

In general, a formula is valid if it is valid for all models. However, because the worlds are closely related to policies, not all models are reasonable models given how policies relate to each other. Becker et al. defined TM models, where each TM model models all possible policies and all possible policy interactions. One important interaction between policies are the relative strength of policies. This is what we use in our conversion rule as well. One interesting observation is that our □R rule only picks one world that is accessible from w given γ, namely (w, γ). The Kripke semantics require φ hold in all worlds related to w given γ ($R_{|\gamma|_M}(w, w')$). This is another reason why we need the conversion rule. Intuitively, this rule makes sure that checking φ true in one accessible world is good enough. This also suggests that we could modify our □R to build in conversion so all accessible worlds are considered. We leave this for future work.

New judgement in the premise The PolCut and Conv rules introduce a new judgement $p\ @\ w$ into the premise. As a consequence, our sequent calculus does not have the traditional sub-formula property. However, because we only consider systems with a finite number of atomic propositions, there is only a finite number of such $p\ @\ w$. Further, some of the proofs of the meta-properties (e.g., soundness) rely on the fact that the new judgment $p\ @\ w$ is only introduced when it is derivable: in the case of PolCut, $w \Vdash_d p$; and in the case of Conv, Γ already includes $p\ @\ w'$ where w' is weaker than w. If we were to allow the general cut rule, those proofs would have been very difficult, if not impossible.

Applications to Other Domains In addition to security, Datalog has been widely used in many other domains. For instance, Datalog has been used to model networks and diagnose network configuration [14, 24]. The counterfactual reasoning framework could potentially be applied to these domains to diagnose and pinpoint specific configurations or nodes in the network such that if it weren't for those nodes, the error wouldn't have occurred.

Counterfactual reasoning is a meta-level reasoning about a system. Here, this system is Horn clauses. The derivability relation for Horn clauses is fairly simple, so it is a great starting point to build a

formal proof system for counterfactual reasoning. We are intersted in generalizing this framework to reason about counterfactuals of other logic, for instance, a constructive authorization logic. We can still interpret $\Box_\gamma \varphi$ as by submitting additional rules γ, φ is derivable. Then the challenge is how to design proof rules to capture the counterfactual reasoning in the same way the Dlog rule does for Horn clauses. Another challenge is to identify the requirements of the worlds in the Kripke structure to make it compatible with the semantics of that logic.

7 RELATED WORK

Counterfactual reasoning has been studied by philosophers for decades [10, 22, 26] after Lewis's seminal book was published in 1973 [22]. Most closely related to our reading of the counterfactual conditions $\Box_\gamma \varphi$ is work on conditional logic [2, 3, 11, 17, 18, 21, 29, 30]. In conditional logic, $H > F$ is read as: if H were true then F would have been the case; or F follows from H in all worlds where H is true. $\Box_\gamma \varphi$ is similar to $\gamma > \varphi$. There is a set of axioms dictating how conditionals interact with the rest of logic. Each conditional logic admits a different set of axioms. Most of the axioms that our sequent calculus support can be mapped to axioms of conditional logic. One exception is the Dlog axiom, which is intrinsic to Horn clause reasoning. Conditional logic is given possible world semantics. Similar to the Kripke semantics that Becker et al. introduced, there is a notion of accessibility of worlds indexed by conditionals (H), and a notion of a minimal world or multiple minimal worlds that are accessible from w with distance H. Because we target Horn clauses, the accessibility relation of the Kripke semantics for our logic is customized to accommodate properties of Horn clauses. To some extent, our proof system can be viewed as a special instance of a more general conditional logic. The concrete semantics help us avoid many of the unsatisfactory explanations of axioms in a more general conditional logic.

Only relatively recently, applications of counterfactual and conditional reasoning have found their way into computer science. Samet [36] developed a reasoning system based on conditional logic for game theory. Players deduce their next moves and backtrack possible states by considering other possible worlds. Halpern later reformalized Samet's system by using a combination of epistemic and conditional logic [19]. Becker et al. used counterfactual reasoning to model probing attacks in trust management systems [5].

Our work is inspired by the axiomatized proof system developed by Becker et al. [5]. Unfortunately, axiomatized proof systems are known to be difficult for proof construction. To address this problem, most recently, Pasarella et al. used a logic programming language [27] as the operational framework for constructing proofs in counterfactual reasoning in trust management system [31]. Their extended semantics for logic programs, written $P \vdash_{\hat{O}} G$, model semantics of counterfactual conditions. The validity of a formula G satisfies the condition that it is not possible to find a policy Δ such that $\Delta \vdash_{\hat{O}} \neg G$. Different from our work, Pasarella and Lobo's do not provide sequent calculus formulation for counterfactual reasoning.

We believe we are the first to define a sequent calculus for counterfactual conditionals based on Horn clauses. Our sequent calculus has cut elimination. All proofs of our metatheories are clean and easy to follow as they are either by induction over the structure of the formula, or by induction over the sequent calculus rules.

Our formalization is influenced by the formalization of classical modal S5 by Murphy et al. [28] and work on judgmental reconstruction of modal logic [34]. Here, the counterfactual conditional γ is treated as a world in the Kripke model and therefore, we can borrow ideas from sequent calculus for modal logic. Interestingly, our initial attempt that does not use a separate judgment φ @ w failed in very similar ways as Prawitz's formulations of necessity [35].

8 CONCLUSION

We presented a sequent calculus for counterfactual reasoning of Horn clauses, which can model probing attacks in trust management systems concisely. Our sequent calculus has cut elimination [32] and is sound and complete with respect to the axiomatized system proposed by Becker et al. We plan to explore applications of our sequent calculus to model counterfactual reasonings in domains including security, networks, and program analysis.

ACKNOWLEDGEMENT

This research was supported in part by NSF grant CNS1423168.

REFERENCES

[1] Serge Abiteboul, Richard Hull, and Victor Vianu. 1994. *Foundations of Databases.* Pearson, Cambridge, MA, USA.
[2] Régis Alenda, Nicola Olivetti, and Gian Luca Pozzato. 2012. Nested Sequent Calculi for Conditional Logics. In *Logics in Artificial Intelligence (Lecture Notes in Computer Science)*, Vol. 7519. Springer, 14–27.
[3] Horacio Arlo-Costa. 2014. The Logic of Conditionals. In *The Stanford Encyclopedia of Philosophy* (summer 2014 ed.), Edward N. Zalta (Ed.).
[4] Moritz Becker, Cedric Fournet, and Andrew Gordon. 2007. Design and Semantics of a Decentralized Authorization Language. In *Proceedings of the 20th IEEE Computer Security Foundations Symposium (CSF'07).*
[5] Moritz Y Becker, Alessandra Russo, and Nik Sultana. 2012. Foundations of logic-based trust management. In *Proceedings of the 2012 IEEE Symposium on Security and Privacy (SP '12).*
[6] Matt Blaze, Joan Feigenbaum, and Jack Lacy. 1996. Decentralized Trust Management. In *Proceedings of the 1996 IEEE Symposium on Security and Privacy (SP '96).*
[7] Piero A. Bonatti. 2011. Datalog for Security, Privacy and Trust. In *Proceedings of the First International Conference on Datalog Reloaded (Datalog'10).*
[8] Martin Bravenboer and Yannis Smaragdakis. 2009. Strictly Declarative Specification of Sophisticated Points-to Analyses. In *Proceedings of the 24th ACM SIGPLAN Conference on Object Oriented Programming Systems Languages and Applications (OOPSLA '09).*
[9] Michael Burrows, Martin Abadi, and Roger Needham. 1990. A Logic of Authentication. *ACM Trans. Comput. Syst.* 8, 1 (Feb. 1990).
[10] J. Collins, E. Hall, and L. Paul. 2004. *Causation and Counterfactuals.* MIT Press, Cambridge, Mass.
[11] H C M de Swart. 1983. Gentzen- or Beth-Type System, a Practical Decision Procedure and a Constructive Completeness Proof for the Counterfactual Logics VC and VCS. *The Journal of Symbolic Logic* 48, 1 (March 1983), 1–20.
[12] John DeTreville. 2002. Binder, a Logic-Based Security Language. In *Proceedings of the 2002 IEEE Symposium on Security and Privacy (SP '02).*
[13] Henry DeYoung, Deepak Garg, and Frank Pfenning. 2008. An Authorization Logic With Explicit Time. In *Proceedings of the 2008 21st IEEE Computer Security Foundations Symposium (CSF '08).*
[14] Ari Fogel, Stanley Fung, Luis Pedrosa, Meg Walraed-Sullivan, Ramesh Govindan, Ratul Mahajan, and Todd Millstein. 2015. A General Approach to Network Configuration Analysis. In *Proceedings of the 12th USENIX Conference on Networked Systems Design and Implementation (NSDI'15).*
[15] Deepak Garg and Martín Abadi. 2008. A Modal Deconstruction of Access Control Logics. In *Proceedings of the 11th International Conference on Foundations of Software Science and Computational Structures (FOSSACS'08).*
[16] Valerio Genovese, Laura Giordano, Valentina Gliozzi, and Gian Luca Pozzato. 2011. A Conditional Constructive Logic for Access Control and Its Sequent Calculus. In *Automated Reasoning with Analytic Tableaux and Related Methods (Lecture Notes in Computer Science)*, Vol. 6793. Springer, 164–179.

[17] Ian Philip Gent. 1992. A Sequent- or Tableau-Style System for Lewis's Counterfactual Logic VC. *Notre Dame Journal of Formal Logic* 33, 3 (1992), 369–382.

[18] Laura Giordano, Valentina Gliozzi, Nicola Olivetti, and Camilla Schwind. 2009. Tableau Calculus for Preference-Based Conditional Logics: PCL and its Extensions. *ACM Transactions on Computational Logic* 10, 3 (April 2009), 1–50.

[19] Joseph Y. Halpern. 2000. Hypothetical Knowledge and Counterfactual Reasoning. In *Very Large Data Bases*.

[20] Trevor Jim. 2001. SD3: A Trust Management System with Certified Evaluation. In *Proceedings of the 2001 IEEE Symposium on Security and Privacy (SP'01)*.

[21] Björn Lellmann and Dirk Pattinson. 2012. Sequent Systems for Lewis' Conditional Logics. In *Logics in Artificial Intelligence (Lecture Notes in Computer Science)*, Vol. 7519. Springer, 320–332.

[22] David Lewis. 1973. *Counterfactuals*. Blackwell, Oxford.

[23] Ninghui Li, Benjamin Grosof, and Joan Feigenbaum. 2000. A Practically Implementable and Tractable Delegation Logic. In *Proceedings of the 2000 IEEE Symposium on Security and Privacy (SP '00)*.

[24] Boon Thau Loo, Tyson Condie, Minos Garofalakis, David E. Gay, Joseph M. Hellerstein, Petros Maniatis, Raghu Ramakrishnan, Timothy Roscoe, and Ion Stoica. 2006. Declarative Networking: Language, Execution and Optimization. In *Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data (SIGMOD '06)*.

[25] McKenna McCall, Lay Kuan Loh, and Limin Jia. 2017. *A Sequent Calculus for Counterfactual Reasoning*. Technical Report CMU-CyLab-17-003. Cylab, Carnegie Mellon University.

[26] Peter Menzies. 2014. Counterfactual Theories of Causation. In *The Stanford Encyclopedia of Philosophy* (spring 2014 ed.), Edward N. Zalta (Ed.).

[27] Dale Miller. 1989. A Logical Analysis of Modules in Logic Programming. *The Journal of Logic Programming* 6, 1-2 (Jan. 1989), 79–108.

[28] Tom Murphy, Karl Crary, and Robert Harper. 2005. Distributed Control Flow with Classical Modal Logic. In *Computer Science Logics (Lecture Notes in Computer Science)*, Vol. 3634. Springer, 51–69.

[29] Nicola Olivetti and Gian Luca Pozzato. 2015. A Standard Internal Calculus for Lewis' Counterfactual Logics. In *Proceedings of the 24th International Conference on Automated Reasoning with Analytic Tableaux and Related Methods - Volume 9323 (TABLEAUX 2015)*.

[30] Nicola Olivetti, Gian Luca Pozzato, and Camilla B Schwind. 2007. A Sequent Calculus and a Theorem Prover for Standard Conditional Logics. *ACM Transactions on Computational Logic* 8, 4 (August 2007).

[31] Edelmira Pasarella and Jorge Lobo. 2015. Reasoning about Policy Behavior in Logic-Based Trust Management Systems: Some Complexity Results and an Operational Framework. In *Proceedings of the 28th IEEE Computer Security Foundations Symposium (CSF'15)*.

[32] Frank Pfenning. 1995. Structural Cut Elimination. In *Proceedings of the 10th Annual IEEE Symposium on Logic in Computer Science (LICS '95)*.

[33] Frank Pfenning. 2000. Structural Cut Elimination: I. Intuitionistic and Classical Logic. *Information and Computation* 157, 1-2 (February 2000), 84–141.

[34] Frank Pfenning and Rowan Davies. 2001. A Judgmental Reconstruction of Modal Logic. *Mathematical Structures in Computer Science* 11, 4 (Aug. 2001), 511–540.

[35] D Prawitz. 1965. *Natural Deduction*. Almquist Wiksell, Stockholm.

[36] D. Samet. 1996. Hypothetical Knowledge and Games with Perfect Information. *Games and Economic Behavior* 17 (1996), 230–251.

[37] Leon Sterling and Ehud Shapiro. 1986. *The Art of Prolog: Advanced Programming Techniques*. MIT Press, Cambridge, MA, USA.

[38] Ankur Taly, Úlfar Erlingsson, John C. Mitchell, Mark S. Miller, and Jasvir Nagra. 2011. Automated Analysis of Security-Critical JavaScript APIs. In *Proceedings of the 2011 IEEE Symposium on Security and Privacy (SP '11)*.

A LEMMAS AND PROOFS FOR SOUNDNESS

Proving Lemma 7 is fairly involved. Inducting over the structure of $\mathcal{E} :: \gamma_1 \ @ \ \top \vdash_\Sigma \gamma_2 \ @ \ \top$ proves insufficient because PolCut and Conv can introduce judgements that our induction hypothesis does not account for. We have to prove a more general lemma instead. First, we define a new calculus which contains labels that will allow us to track judgements which have been introduced by, or are derived from judgements introduced by, PolCut and Conv. This will allow us to distinguish judgements of the form $\gamma \ @ \ \top$ from those of the form $p \ @ \ w$. We assign the label 'A' to judgements that have been added by, or are derived from judgements which have been added by, PolCut$^+$ or Conv$^+$. All other judgements are labeled with 'o'.

Our syntax is the same as in Section 4 with a few exceptions shown below.

$$\begin{array}{lrcl} \textit{Label} & x & ::= & A \mid o \\ \textit{Judgement} & J, (J)_x & ::= & \gamma \ @ \ w \mid (\gamma \ @ \ w)_x \\ \textit{Contexts} & \Gamma^+ & ::= & \cdot \mid \Gamma^+, (J)_x \end{array}$$

The judgment for the marked up derivation is $\Gamma^+ \vdash^+_\Sigma \Delta$. The sequent rules are shown in Figure 9. For the following lemmas, we define some notation:

$$\begin{array}{ll} \forall J \in \Gamma^+_p, J = (q \ @ \ w)_A & \forall J \in \Gamma_p, J = q \ @ \ w \\ \forall J \in \Gamma^+_\top, J = (\gamma \ @ \ \top)_o & \forall J \in (\Gamma_\top, \Delta_\top), J = \gamma \ @ \ \top \\ \forall J \in \Gamma^+_{\mathbf{At}}, J = (p \ @ \ \vec{q})_o & \forall J \in (\Gamma_{\mathbf{At}}, \Delta_{\mathbf{At}}), J = p \ @ \ \vec{q} \end{array}$$

We prove that derivations using original sequent rules can be annotated, which result in derivations in newly defined sequent rules.

Lemma 16. *If* $\mathcal{E} :: \Gamma_p, \Gamma_\top, \Gamma_{\mathbf{At}} \vdash_\Sigma \Delta_\top, \Delta_{\mathbf{At}}$ *then* $\exists \mathcal{D} :: \Gamma^+_p, \Gamma^+_\top, \Gamma^+_{\mathbf{At}} \vdash^+_\Sigma \Delta_\top, \Delta_{\mathbf{At}}$

Proof (sketch): By induction over the structure of $\mathcal{E}$. Most of the work for this proof involves showing that the induction hypothesis can be applied to the subderivations of $\mathcal{E}$. Once that has been shown, the conclusion follows from applying the same rule from the labeled version of the sequent calculus on the result(s) of applying the induction hypothesis. □

Next, we relate derivations in the marked up sequent rules to formula semantics. We define polOf(J) to transform a judgment to a policy as follows.

$$\begin{array}{lcl} \text{polOf}(\gamma \ @ \ \top) & = & \gamma \\ \text{polOf}(p \ @ \ \vec{q}) & = & p \text{ :- } \vec{q} \end{array}$$

Lemma 17. *If* $\mathcal{E} :: \Gamma^+_p, \Gamma^+_\top, \Gamma^+_{\mathbf{At}} \vdash^+_\Sigma \Delta_\top, \Delta_{\mathbf{At}}$ *where* $\forall J \in \Gamma^+_p, J = q \ @ \ w$ *and polOf* $(\Gamma^+_\top, \Gamma^+_{\mathbf{At}}, w) \Vdash q$, *then* $\exists J \in (\Delta_\top, \Delta_{\mathbf{At}})$ *s.t.* *polOf* $(\Gamma^+_\top, \Gamma^+_{\mathbf{At}}) \Vdash$ *polOf* (J)

Proof (sketch): By induction over the structure of $\mathcal{E}$. Most of the work for this proof involves showing that the induction hypothesis can be applied to the subderivations of $\mathcal{E}$. Then, the conclusion follows directly, or from one of the lemmas from Appendix B. □

Lemma 7. $\forall \gamma_1, \gamma_2$, *If* $\mathcal{E} :: \gamma_1 \ @ \ \top \vdash_\Sigma \gamma_2 \ @ \ \top$ *then* $\overline{\gamma_1} \Vdash \gamma_2$.

Proof (sketch): The proof follows from the observation that $\mathcal{E}$ is of the form $\Gamma_\top \vdash_\Sigma \Delta_\top$. Then the conclusion follows from, Lemma 16 and Lemma 17. □

B PROPERTIES OF $\Vdash$ AND $\Vdash_d$

Lemma 18 (Admissibility of Cut for $\Vdash$)**.** *If* $\mathcal{D} :: w \Vdash \gamma_1$ *and* $\mathcal{E} :: w, \overline{\gamma_1} \Vdash \gamma_2$ *then* $\exists \mathcal{F} :: w \Vdash \gamma_2$

Proof (sketch): If we consider the definitions for $\Vdash$ as sequent, this is a proof of admissibility of cut. So we proceed by induction first over the structure of γ_1 and then over the structures of $\mathcal{D}$ and $\mathcal{E}$. The proof closely resembles our proof of Theorem 4 and requires weakening, contraction for $\Vdash$. Most cases are straightforward, except for the one where $\mathcal{D}$ ends in $\Box$R and $\mathcal{E}$ ends in At which requires Lemma 23. We also need Lemma 22 to prove the admissibility of cut for $\Vdash_d$. □

$$\frac{}{\Gamma^+, (p \mathbin{@} w)_x \vdash^+_\Sigma p \mathbin{@} w, \Delta}\ \textsc{Init}^+ \qquad \frac{}{\Gamma^+ \vdash^+_\Sigma \top \mathbin{@} w, \Delta}\ \top\,\mathrm{R}^+ \qquad \frac{\Gamma^+ \vdash^+_\Sigma \gamma_1 \mathbin{@} w, \gamma_1 \wedge \gamma_2 \mathbin{@} w, \Delta \qquad \Gamma^+ \vdash^+_\Sigma \gamma_2 \mathbin{@} w, \gamma_1 \wedge \gamma_2 \mathbin{@} w, \Delta}{\Gamma^+ \vdash^+_\Sigma \gamma_1 \wedge \gamma_2 \mathbin{@} w, \Delta}\ \wedge\mathrm{R}^+$$

$$\frac{\Gamma^+, (\gamma_1 \mathbin{@} w)_o, (\gamma_1 \wedge \gamma_2 \mathbin{@} w)_o, \vdash^+_\Sigma \Delta}{\Gamma^+, (\gamma_1 \wedge \gamma_2 \mathbin{@} w)_o \vdash^+_\Sigma \Delta}\ \wedge\mathrm{L1}^+ \qquad \frac{\Gamma^+, (\gamma_2 \mathbin{@} w)_o, (\gamma_1 \wedge \gamma_2 \mathbin{@} w)_o, \vdash^+_\Sigma \Delta}{\Gamma^+, (\gamma_1 \wedge \gamma_2 \mathbin{@} w)_o \vdash^+_\Sigma \Delta}\ \wedge\mathrm{L2}^+ \qquad \frac{\Gamma^+ \vdash^+_\Sigma \varphi \mathbin{@} (w, \overline{\gamma}), \Box_\gamma \varphi \mathbin{@} w, \Delta}{\Gamma^+ \vdash^+_\Sigma \Box_\gamma \varphi \mathbin{@} w, \Delta}\ \Box\mathrm{R}^+$$

$$\frac{\Gamma^+, (\Box_\gamma \varphi \mathbin{@} w)_x, (\varphi \mathbin{@} (w, \overline{\gamma}))_x \vdash^+_\Sigma \Delta}{\Gamma^+, (\Box_\gamma \varphi \mathbin{@} w)_x \vdash^+_\Sigma \Delta}\ \Box\mathrm{L}^+ \qquad \frac{w \Vdash_d p \qquad \mathsf{symbols}(p \mathbin{@} w) \subseteq \Sigma \qquad \Gamma^+, (p \mathbin{@} w)_A \vdash^+_\Sigma \Delta}{\Gamma^+ \vdash^+_\Sigma \Delta}\ \textsc{PolCut}^+$$

$$\frac{\Gamma^+, (p \mathbin{@} (w, w', q))_x, (q \mathbin{@} w)_y, (p \mathbin{@} (w, w'))_x \vdash^+_\Sigma \Delta}{\Gamma^+, (p \mathbin{@} (w, w', q))_x, (q \mathbin{@} w)_y \vdash^+_\Sigma \Delta}\ \textsc{PolL-W-At}^+$$

$$\frac{\Gamma^+, (p \mathbin{@} (w, w', \Box_{\vec{a}}\, b))_A, (b \mathbin{@} (w, \vec{a}))_x, (p \mathbin{@} (w, w'))_A \vdash^+_\Sigma \Delta}{\Gamma^+, (p \mathbin{@} (w, w', \Box_{\vec{a}}\, b))_A, (b \mathbin{@} (w, \vec{a}))_x \vdash^+_\Sigma \Delta}\ \textsc{PolL-W-}\Box^+$$

$$\frac{\Gamma^+, (s \mathbin{@} (q :\!\!- p_1, \cdots, p_n, w))_A, (s \mathbin{@} w)_A \vdash^+_\Sigma \Delta \qquad \Gamma^+, (s \mathbin{@} (q :\!\!- p_1, \cdots, p_n, w))_A, (s \mathbin{@} (q, w))_A, (p_1 \mathbin{@} w)_A, \cdots, (p_n \mathbin{@} w)_A \vdash^+_\Sigma \Delta}{\Gamma^+, (s \mathbin{@} (q :\!\!- p_1, \cdots, p_n, w))_A \vdash^+_\Sigma \Delta}\ \textsc{Dlog}^+$$

$$\frac{\mathsf{symbols}(w_2) \subseteq \Sigma \qquad \Gamma^+, (p \mathbin{@} w_1)_x, (p \mathbin{@} w_2)_A \vdash^+_\Sigma \Delta \qquad (\wedge(w_2) \mathbin{@} \top)_o \vdash^+_\Sigma \wedge(w_1) \mathbin{@} \top}{\Gamma^+, (p \mathbin{@} w_1)_x \vdash^+_\Sigma \Delta}\ \textsc{Conv}^+$$

Figure 9: Labeled sequent calculus

Lemma 19. *$\forall w$, If $\mathcal{E} :: w, p :\!\!- q_1 \cdots q_n \Vdash_d s$ then*
(1) $\exists \mathcal{D} :: w \Vdash_d s$, or
(2) $\exists \mathcal{D} :: w, p \Vdash_d s$, and $\forall i \in [1, n]\ \exists \mathcal{D}_i :: w \Vdash_d q_i$

Proof (sketch): By induction over the structure of $\mathcal{E}$. When $\mathcal{E}$ ends in At, the conclusion follows directly from the At rule.
When $\mathcal{E}$ ends in Dlog and $p = s$, the conclusion follows from At and the induction hypothesis. Otherwise, when $p \neq s$, the conclusion follows from the induction hypothesis and Dlog.

□

Lemma 20 (Contraction for $\Vdash$). *$\forall w, w', \gamma$ If $\mathcal{E} :: w, w', w' \Vdash \gamma$ then $\exists \mathcal{D} :: w, w' \Vdash \gamma$*

Proof (sketch): By induction over the structure of $\mathcal{E}$. We can get the conclusion by applying the same rule on the result of using the induction hypothesis. Lemma 24 is required to prove contraction for $\Vdash_d$ □

Lemma 21 (Weakening for $\Vdash$). *$\forall w, w', \gamma$ If $\mathcal{E} :: w' \Vdash \gamma$ then $\exists \mathcal{D} :: w, w' \Vdash \gamma$ with $|\mathcal{D}| = |\mathcal{E}|$*

Proof (sketch): By induction over the structure of $\mathcal{E}$. We can get the conclusion by applying the same rule on the result of using the induction hypothesis. Lemma 25 is required to prove weakening for $\Vdash_d$ □

Lemma 22 (Admissibility of Cut for $\Vdash_d$). *$\forall w$ If $\mathcal{D} :: w \Vdash_d p$ and $\mathcal{E} :: w, p \Vdash_d q$ then $\exists \mathcal{F} :: w \Vdash q$*

Proof (sketch): By nested induction on the structures of $\mathcal{D}$ and $\mathcal{E}$. The conclusion follows from contraction when $\mathcal{D}$ ends in At and $\mathcal{E}$ is arbitrary and from the At rule when $\mathcal{D}$ is arbitrary and $\mathcal{E}$ ends in At. The last case, when $\mathcal{D}$ and $\mathcal{E}$ both end in Dlog, follows from applying Dlog on the result of applying the induction hypothesis on $\mathcal{D}$ and the subderivations of $\mathcal{E}$. □

Lemma 23. *$\forall w$ If $\mathcal{D} :: w, \vec{q} \Vdash_d r$ and $\mathcal{E} :: w, \Box_{\vec{q}}\, r \Vdash_d p$ then $\exists \mathcal{F} :: w \Vdash_d p$*

Proof (sketch): By nested induction over the structures of $\mathcal{D}$ and $\mathcal{E}$. The conclusion follows from the At rule when $\mathcal{D}$ is arbitrary and $\mathcal{E}$ ends in At. When $\mathcal{D}$ is arbitrary and $\mathcal{E}$ ends in Dlog, the conclusion follows from applying Dlog on the results of applying the induction hypothesis on $\mathcal{D}$ and the subderivations of $\mathcal{E}$. □

Lemma 24 (Contraction for $\Vdash_d$). *$\forall w, w', \gamma$ If $\mathcal{E} :: w, w', w' \Vdash_d \gamma$ then $\exists \mathcal{D} :: w, w' \Vdash_d \gamma$*

Proof (sketch): By induction over the structure of $\mathcal{E}$. When $\mathcal{E}$ ends in At, the conclusion follows directly from the At rule. When $\mathcal{E}$ ends in Dlog, we can get the conclusion by applying the Dlog on the result of using the induction hypothesis on the subderivations of $\mathcal{E}$. □

Lemma 25 (Weakening for $\Vdash_d$). *$\forall w, w', \gamma$ If $\mathcal{E} :: w' \Vdash_d \gamma$ then $\exists \mathcal{D} :: w, w' \Vdash_d \gamma$ with $|\mathcal{D}| = |\mathcal{E}|$*

Proof (sketch): By induction over the structure of $\mathcal{E}$. When $\mathcal{E}$ ends in At, the conclusion follows directly from the At rule. When $\mathcal{E}$ ends in Dlog, we can get the conclusion by applying the Dlog on the result of using the induction hypothesis on the subderivations of $\mathcal{E}$. □

C ADMISSIBILITY OF CUT

Theorem 4 (Admissibility of Cut).
If $\mathcal{D} :: \Gamma \vdash \varphi \ @ \ w, \Delta$ and $\mathcal{E} :: \Gamma, \varphi \ @ \ w \vdash \Delta$,
then $\exists \mathcal{F} :: \Gamma \vdash \Delta$

Proof.
We show some representative cases below. The ones not shown proceed in the same fashion.

Base cases
Case: $\mathcal{D}$ ends in Init and $\mathcal{E}$ is arbitrary.

$$\mathcal{D} = \frac{}{\Gamma, p \ @ \ w \vdash_\Sigma p \ @ \ w, \Delta}\ \textsc{Init}$$
and $\mathcal{E} :: \Gamma, p \ @ \ w, p \ @ \ w \vdash_\Sigma \Delta$

To show: $\exists \mathcal{F} :: \Gamma, p \ @ \ w \vdash_\Sigma \Delta$
$\mathcal{F}$ may be obtained by applying the contraction lemma on $\mathcal{E}$.
Case: $\mathcal{D}$ is arbitrary and $\mathcal{E}$ ends in Init.

$\mathcal{D} :: \Gamma \vdash_\Sigma p \ @ \ w, p \ @ \ w, \Delta$
and $\mathcal{E} = \dfrac{}{\Gamma, p \ @ \ w \vdash_\Sigma p \ @ \ w, \Delta}\ \textsc{Init}$

To show: $\exists \mathcal{F} :: \Gamma \vdash_\Sigma p \ @ \ w, \Delta$
$\mathcal{F}$ may be obtained by applying the contraction lemma on $\mathcal{D}$.
Case: $\mathcal{D}$ ends in $\top$ R and $\mathcal{E}$ is arbitrary.
$\top \ @ \ w$ will be a side formula in $\mathcal{E}$.
Case: $\mathcal{D}$ is arbitrary and $\mathcal{E}$ ends in $\top$ R.
The cut formula is a side formula in $\mathcal{E}$.
Case: $\mathcal{D}$ ends in $\bot$ L and $\mathcal{E}$ is arbitrary.
The cut formula is a side formula in $\mathcal{D}$.
Case: $\mathcal{D}$ is arbitrary and $\mathcal{E}$ ends in $\bot$ L.
$\bot \ @ \ w$ is a side formula in $\mathcal{D}$

Principal cases
Case: $\mathcal{D}$ ends in $\Box$ R and $\mathcal{E}$ ends in $\Box$ L

$$\mathcal{D} = \frac{\mathcal{D}_1 :: \Gamma \vdash_\Sigma \varphi \ @ \ (w, \overline{\gamma}), \Box_\gamma \varphi \ @ \ w, \Delta}{\Gamma \vdash_\Sigma \Box_\gamma \varphi \ @ \ w, \Delta}\ \Box\text{R}$$

$$\mathcal{E} = \frac{\mathcal{E}_1 :: \Gamma, \Box_\gamma \varphi \ @ \ w, \varphi \ @ \ (w, \overline{\gamma}) \vdash_\Sigma \Delta}{\Gamma, \Box_\gamma \varphi \ @ \ w \vdash_\Sigma \Delta}\ \Box\text{L}$$

(G) To show: $\exists \mathcal{F} :: \Gamma \vdash_\Sigma \Delta$
Weakening and IH on $\Box_\gamma \varphi \ @ \ w$, $\mathcal{D}$, and $\mathcal{E}_1$ gives
(1) $\mathcal{E}_1' :: \Gamma, \varphi \ @ \ (w, \overline{\gamma}) \vdash_\Sigma \Delta$
Weakening and IH on $\Box_\gamma \varphi \ @ \ w$, $\mathcal{D}_1$, and $\mathcal{E}$ gives
(2) $\mathcal{D}_1' :: \Gamma \vdash_\Sigma \varphi \ @ \ (w, \overline{\gamma}), \Delta$
IH on $\varphi \ @ \ (w, \overline{\gamma})$, $\mathcal{D}_1'$, and $\mathcal{E}_1'$ gives (G)

Special cases
These cases are already covered by the base cases and side cases.

We show one case in more detail, here, to make this explicit.
Case: $\mathcal{E}$ ends in Conv
Subcase: $\mathcal{D}$ ends in Init

$$\mathcal{D} = \frac{}{\Gamma, p \ @ \ w_1 \vdash_\Sigma p \ @ \ w_1, \Delta}\ \textsc{Init}$$

$$\mathcal{E} = \frac{\begin{array}{c}\mathcal{E}_1 :: \text{symbols}(w_2) \subseteq \Sigma \\ \mathcal{E}_2 :: \Gamma, p \ @ \ w_1, p \ @ \ w_1, p \ @ \ w_2 \vdash_\Sigma \Delta \\ \mathcal{E}_3 :: \wedge(w_2) \ @ \ \top \vdash_\Sigma \wedge(w_1) \ @ \ \top\end{array}}{\Gamma, p \ @ \ w_1, p \ @ \ w_1 \vdash_\Sigma \Delta}\ \textsc{Conv}$$

To show: $\exists \mathcal{F} :: \Gamma, p \ @ \ w_1 \vdash_\Sigma \Delta$
Applying the contraction lemma on $\mathcal{E}$ gives $\mathcal{F}$

The rest of the subcases are similar. We show one below.
Subcase: $\mathcal{D}$ ends in $\vee$ R1
Observe that $p \ @ \ w_1$ must be a side formula in $\mathcal{D}$.

$$\mathcal{D} = \frac{\mathcal{D}_1 :: \Gamma \vdash_\Sigma p \ @ \ w_1, \varphi_1 \ @ \ w, \varphi_1 \vee \varphi_2 \ @ \ w, \Delta}{\Gamma \vdash_\Sigma p \ @ \ w_1, \varphi_1 \vee \varphi_2 \ @ \ w, \Delta}\ \vee\text{R1}$$

$$\mathcal{E} = \frac{\begin{array}{c}\mathcal{E}_1 :: \text{symbols}(w_2) \subseteq \Sigma \\ \mathcal{E}_2 :: \Gamma, p \ @ \ w_1, p \ @ \ w_2 \vdash_\Sigma \varphi_1 \vee \varphi_2 \ @ \ w, \Delta \\ \mathcal{E}_3 :: \wedge(w_2) \ @ \ \top \vdash_\Sigma \wedge(w_1) \ @ \ \top\end{array}}{\Gamma, p \ @ \ w_1 \vdash_\Sigma \varphi_1 \vee \varphi_2 \ @ \ w, \Delta}\ \textsc{Conv}$$

(G) To show: $\exists \mathcal{F} :: \Gamma \vdash_\Sigma \varphi_1 \vee \varphi_2 \ @ \ w, \Delta$
Apply weakening and IH on $p \ @ \ w_1$, $\mathcal{D}_1$, and $\mathcal{E}$ gives
(1) $\mathcal{D}_1' :: \Gamma \vdash_\Sigma \varphi_1 \ @ \ w, \varphi_1 \vee \varphi_2 \ @ \ w, \Delta$
Applying $\vee$R1 on $\mathcal{D}_1'$ gives (G)

Side cases on $\mathcal{D}$
These cases are all similar to each other.
We show all of the cases for our new rules, except $\Box$R, $\Box$L, and PolL-W-$\Box$ plus one involving Init.
Case: $\mathcal{D}$ ends in Init and $\mathcal{E}$ is arbitrary

$$\mathcal{D} = \frac{}{\Gamma, p \ @ \ w \vdash_\Sigma \varphi' \ @ \ w', p \ @ \ w, \Delta}\ \textsc{Init}$$
and $\mathcal{E} :: \Gamma, p \ @ \ w, \varphi' \ @ \ w' \vdash_\Sigma p \ @ \ w, \Delta$

To show: $\exists \mathcal{F} :: \Gamma, p \ @ \ w \vdash_\Sigma p \ @ \ w, \Delta$
Let $\mathcal{F} = \dfrac{}{\Gamma, p \ @ \ w \vdash_\Sigma p \ @ \ w, \Delta}\ \textsc{Init}$
Case: $\mathcal{D}$ ends in PolCut and $\mathcal{E}$ is arbitrary

$$\mathcal{D} = \frac{\begin{array}{c}\mathcal{D}_1 :: w \Vdash_d p \qquad \mathcal{D}_2 :: \text{symbols}(p \ @ \ w) \subseteq \Sigma \\ \mathcal{D}_3 :: \Gamma, p \ @ \ w \vdash_\Sigma \varphi' \ @ \ w', \Delta\end{array}}{\Gamma \vdash_\Sigma \varphi' \ @ \ w', \Delta}\ \textsc{PolCut}$$
and $\mathcal{E} :: \Gamma, \varphi' \ @ \ w' \vdash_\Sigma \Delta$

(G) To show: $\exists \mathcal{F} :: \Gamma \vdash_\Sigma \Delta$
Weakening and IH on $\varphi' \ @ \ w'$, $\mathcal{D}_3$, and $\mathcal{E}$ gives
(1) $\mathcal{D}_3' :: \Gamma, p \ @ \ w \vdash_\Sigma \Delta$
Applying PolCut on $\mathcal{D}_1$, $\mathcal{D}_2$, and $\mathcal{D}_3'$ gives (G)
Case: $\mathcal{D}$ ends in PolL-W-At and $\mathcal{E}$ is arbitrary

$$\mathcal{D} = \dfrac{\begin{array}{c}\mathcal{D}_1 :: \Gamma, p\ @\ (w, w', q), q\ @\ w, \\ p\ @\ (w, w') \vdash_\Sigma \varphi'\ @\ w_1, \Delta\end{array}}{\Gamma, p\ @\ (w, w', q), q\ @\ w \vdash_\Sigma \varphi'\ @\ w_1, \Delta}\ \textsc{PolL-W-At}$$
and $\mathcal{E} :: \Gamma, p\ @\ (w, w', q), q\ @\ w, \varphi'\ @\ w_1 \vdash_\Sigma \Delta$

(G) To show: $\exists \mathcal{F} :: \Gamma, p\ @\ (w, w', q), q\ @\ w \vdash_\Sigma \Delta$
Weakening and IH on $\varphi'\ @\ w_1$, $\mathcal{D}_1$, and $\mathcal{E}$ gives
(1) $\mathcal{D}'_1 :: \Gamma, p\ @\ (w, w', q), q\ @\ w, p\ @\ (w, w') \vdash_\Sigma \Delta$
Applying PolL-W-At on $\mathcal{D}'_1$ gives (G)

Case: $\mathcal{D}$ ends in Dlog and $\mathcal{E}$ is arbitrary

$$\mathcal{D} = \dfrac{\begin{array}{c}\mathcal{D}_1 :: \Gamma, s\ @\ (q :\!\text{-}\ p_1, \cdots, p_n, w), s\ @\ w \vdash_\Sigma \varphi'\ @\ w', \Delta \\ \mathcal{D}_2 :: \Gamma, s\ @\ (q :\!\text{-}\ p_1, \cdots, p_n, w), s\ @\ (q, w), \\ p_1\ @\ w, \cdots, p_n\ @\ w \vdash_\Sigma \varphi'\ @\ w', \Delta\end{array}}{\Gamma, s\ @\ (q :\!\text{-}\ p_1, \cdots, p_n, w) \vdash_\Sigma \varphi'\ @\ w', \Delta}\ \textsc{Dlog}$$
and $\mathcal{E} :: \Gamma, s\ @\ (q :\!\text{-}\ p_1, \cdots, p_n, w), \varphi'\ @\ w' \vdash_\Sigma \Delta$

(G) To show: $\exists \mathcal{F} :: \Gamma, s\ @\ (q :\!\text{-}\ p_1, \cdots, p_n, w) \vdash_\Sigma \Delta$
Weakening and IH on $\varphi'\ @\ w'$, $\mathcal{D}_1$, and $\mathcal{E}$ gives
(1) $\mathcal{D}'_1 :: \Gamma, s\ @\ (q :\!\text{-}\ p_1, \cdots, p_n, w), s\ @\ w \vdash_\Sigma \Delta$
Weakening and IH on $\varphi'\ @\ w'$, $\mathcal{D}_2$, and $\mathcal{E}$ gives
(2) $\mathcal{D}'_2 :: \Gamma, s\ @\ (q :\!\text{-}\ p_1, \cdots, p_n, w), s\ @\ (q, w), p_1\ @\ w, \cdots, p_n\ @\ w \vdash_\Sigma \Delta$
Applying Dlog on $\mathcal{D}'_1$ and $\mathcal{D}'_2$ gives (G)

Case: $\mathcal{D}$ ends in Conv and $\mathcal{E}$ is arbitrary

$$\mathcal{D} = \dfrac{\begin{array}{c}\mathcal{D}_1 :: \text{symbols}(w_2) \subseteq \Sigma \\ \mathcal{D}_2 :: \Gamma, p\ @\ w_1, p\ @\ w_2 \vdash_\Sigma \varphi'\ @\ w, \Delta \\ \mathcal{D}_3 :: \wedge(w_2)\ @\ \top \vdash_\Sigma \wedge(w_1)\ @\ \top\end{array}}{\Gamma, p\ @\ w_1 \vdash_\Sigma \varphi'\ @\ w, \Delta}\ \textsc{Conv}$$
and $\mathcal{E} :: \Gamma, p\ @\ w_1, \varphi'\ @\ w \vdash_\Sigma \Delta$

(G) To show: $\exists \mathcal{F} :: \Gamma, p\ @\ w_1 \vdash_\Sigma \Delta$
Weakening and IH on $\varphi'\ @\ w$, $\mathcal{D}_2$, and $\mathcal{E}$ gives
(1) $\mathcal{D}'_2 :: \Gamma, p\ @\ w_1, p\ @\ w_2 \vdash_\Sigma \Delta$
Applying Conv on $\mathcal{D}_1$, $\mathcal{D}'_2$, and $\mathcal{D}_3$ gives (G)

Side cases on $\mathcal{E}$
These cases are all similar to each other.
We show all of the cases for our new rules, except $\Box$R, $\Box$L, and PolL-W-$\Box$, plus one involving Init

Case: $\mathcal{D}$ is arbitrary and $\mathcal{E}$ ends in Init

$$\mathcal{E} = \dfrac{}{\Gamma, p\ @\ w, \varphi'\ @\ w' \vdash_\Sigma p\ @\ w, \Delta}\ \textsc{Init}$$
$\mathcal{D} :: \Gamma, p\ @\ w \vdash_\Sigma \varphi'\ @\ w', p\ @\ w, \Delta$ and

To show: $\exists \mathcal{F} :: \Gamma, p\ @\ w \vdash_\Sigma p\ @\ w, \Delta$
Let $\mathcal{F} = \dfrac{}{\Gamma, p\ @\ w \vdash_\Sigma p\ @\ w, \Delta}\ \textsc{Init}$

Case: $\mathcal{D}$ is arbitrary and $\mathcal{E}$ ends in PolCut

$$\mathcal{E} = \dfrac{\begin{array}{c}\mathcal{E}_1 :: w \Vdash_d p \qquad \mathcal{E}_2 :: \text{symbols}(p\ @\ w) \subseteq \Sigma \\ \mathcal{E}_3 :: \Gamma, p\ @\ w, \varphi'\ @\ w' \vdash_\Sigma \Delta\end{array}}{\Gamma, \varphi'\ @\ w' \vdash_\Sigma \Delta}\ \textsc{PolCut}$$
$\mathcal{D} :: \Gamma \vdash_\Sigma \varphi'\ @\ w, \Delta$ and

(G)To show: $\exists \mathcal{F} :: \Gamma \vdash_\Sigma \Delta$
Weakening and IH on $\varphi'\ @\ w'$, $\mathcal{D}$, and $\mathcal{E}_3$ gives
(1)$\mathcal{E}'_3 :: \Gamma, p\ @\ w \vdash_\Sigma \Delta$
Applying PolCut on $\mathcal{E}_1$, $\mathcal{E}_2$, and $\mathcal{E}'_3$ gives (G)

Case: $\mathcal{D}$ is arbitrary and $\mathcal{E}$ ends in PolL-W-At

$$\mathcal{E} = \dfrac{\begin{array}{c}\mathcal{E}_1 :: \Gamma, p\ @\ (w, w', q), q\ @\ w, p\ @\ (w, w'), \\ \varphi'\ @\ w_1 \vdash_\Sigma \Delta\end{array}}{\Gamma, p\ @\ (w, w', q), q\ @\ w, \varphi'\ @\ w_1 \vdash_\Sigma \Delta}\ \textsc{PolL-W-At}$$
$\mathcal{D} :: \Gamma, p\ @\ (w, w', q), q\ @\ w \vdash_\Sigma \varphi'\ @\ w_1, \Delta$ and

(G)To show: $\exists \mathcal{F} :: \Gamma, p\ @\ (w, w', q), q\ @\ w \vdash_\Sigma \Delta$
Weakening and IH on $\varphi'\ @\ w_1$, $\mathcal{D}$, and $\mathcal{E}_1$ gives
(1)$\mathcal{E}'_1 :: \Gamma, p\ @\ (w, w', q), q\ @\ w, p\ @\ (w, w') \vdash_\Sigma \Delta$
Applying PolL-W-At on $\mathcal{E}'_1$ gives (G)

Case: $\mathcal{D}$ is arbitrary and $\mathcal{E}$ ends in Dlog

$$\mathcal{E} = \dfrac{\begin{array}{c}\mathcal{E}_1 :: \Gamma, s\ @\ (q :\!\text{-}\ p_1, \cdots, p_n, w), s\ @\ w, \varphi'\ @\ w' \vdash_\Sigma \Delta \\ \mathcal{E}_2 :: \Gamma, s\ @\ (q :\!\text{-}\ p_1, \cdots, p_n, w), s\ @\ (q, w), \\ p_1\ @\ w, \cdots, p_n\ @\ w, \varphi'\ @\ w' \vdash_\Sigma \Delta\end{array}}{\Gamma, s\ @\ (q :\!\text{-}\ p_1, \cdots, p_n, w), \varphi'\ @\ w' \vdash_\Sigma \Delta}\ \textsc{Dlog}$$
$\mathcal{D} :: \Gamma, s\ @\ (q :\!\text{-}\ p_1, \cdots, p_n, w) \vdash_\Sigma \varphi'\ @\ w', \Delta$ and

(G)To show: $\exists \mathcal{F} :: \Gamma, s\ @\ (q :\!\text{-}\ p_1, \cdots, p_n, w) \vdash_\Sigma \Delta$
Weakening and IH on $\varphi'\ @\ w'$, $\mathcal{D}$, and $\mathcal{E}_1$ gives
(1)$\mathcal{E}'_1 :: \Gamma, s\ @\ (q :\!\text{-}\ p_1, \cdots, p_n, w), s\ @\ w \vdash_\Sigma \Delta$
Weakening and IH on $\varphi'\ @\ w'$, $\mathcal{D}$, and $\mathcal{E}_2$ gives
(2)$\mathcal{E}'_2 :: \Gamma, s\ @\ (q :\!\text{-}\ p_1, \cdots, p_n, w), s\ @\ (q, w), p_1\ @\ w, \cdots, p_n\ @\ w \vdash_\Sigma \Delta$
Applying Dlog on $\mathcal{E}'_1$ and $\mathcal{E}'_2$ gives (G)

Case: $\mathcal{D}$ is arbitrary and $\mathcal{E}$ ends in Conv

$$\mathcal{E} = \dfrac{\begin{array}{c}\mathcal{E}_1 :: \text{symbols}(w_2) \subseteq \Sigma \\ \mathcal{E}_2 :: \Gamma, p\ @\ w_1, p\ @\ w_2, \varphi'\ @\ w \vdash_\Sigma \Delta \\ \mathcal{E}_3 :: \wedge(w_2)\ @\ \top \vdash_\Sigma \wedge(w_1)\ @\ \top\end{array}}{\Gamma, p\ @\ w_1, \varphi'\ @\ w \vdash_\Sigma \Delta}\ \textsc{Conv}$$
$\mathcal{D} :: \Gamma, p\ @\ w_1 \vdash_\Sigma \varphi'\ @\ w, \Delta$ and

(G)To show: $\exists \mathcal{F} :: \Gamma, p\ @\ w_1 \vdash_\Sigma \Delta$
Weakening and IH on $\varphi'\ @\ w$, $\mathcal{D}$, and $\mathcal{E}_2$ gives
(1)$\mathcal{E}'_2 :: \Gamma, p\ @\ w_1, p\ @\ w_2 \vdash_\Sigma \Delta$
Applying Conv on $\mathcal{E}_1$, $\mathcal{E}'_2$, and $\mathcal{E}_3$ gives (G) □

D PROOFS FOR CONSISTENCY

Theorem 5 (Consistency).
$\forall n, \nexists \mathcal{E}, \nexists w$ such that $\mathcal{E} :: \Gamma \vdash \bot\ @\ w$ and $|\mathcal{E}| \leq n$ where $\forall J \in \Gamma$, J = $p\ @\ w$.

Proof (sketch): By induction on n. When $n = 0$, the conclusion holds by inversion. Assume the conclusion holds when $n < k$. Consider the case when $n = k$. The only possible derivation is one that ends in PolCut, PolL-W-At, PolL-W-$\Box$, Dlog, or Conv because of the restrictions on the judgements in Γ. However, that would yield a proof of $\mathcal{E}' :: \Gamma \vdash \bot\ @\ w'$ and $|\mathcal{E}'| = k - 1$ which contradicts with our assumption that $\nexists \mathcal{E}, \nexists w$ such that $\mathcal{E} :: \Gamma \vdash \bot\ @\ w$ and $|\mathcal{E}| < k$. □

E LEMMAS FOR COMPLETENESS

Lemma 10 (Axiom C1). $\forall \gamma, w, \vdash_\Sigma \gamma\ @\ (\overline{\gamma}, w)$.

Proof (sketch): By induction over the structure of γ. When γ is of the form $\top$, the conclusion follows from the $\top$R rule. For the cases where γ is of the form $p \in \mathbf{At}$ or $q :\!\text{-}\ \vec{p}$, the conclusion follows

from the PolCut rule. Finally, when γ is of the form $\gamma_1 \wedge \gamma_2$, the conclusion follows from of the use of the induction hypothesis on γ_1 and γ_2 and applying $\wedge$R on the weakened result. □

Lemma 26 (Generalized Axiom C2)**.** *For all φ, γ, w, w',*

(1) φ @ $(w, w', \overline{\gamma}), \gamma$ @ $w \vdash_\Sigma \varphi$ @ (w, w') and
(2) φ @ $(w, w'), \gamma$ @ $w \vdash_\Sigma \varphi$ @ $(w, w', \overline{\gamma})$.

Proof (sketch): By induction over the structure of φ. We prove (1) and (2) simultaneously. When φ is of the form $\top$ or $\bot$, the conclusion follows directly from the $\top$R and $\bot$L rules, respectively. When φ is of the form $p \in$ **At**, (1) follows from the PolL-W-At rule and (2) from the Conv rule. The proofs for the rest of the cases where φ involves some connective follow a similar pattern. The conclusion follows from applying the left and right rules for the connective on the result(s) of applying the induction hypothesis on subformulas of φ. □

Lemma 27 (Generalized Axiom Dlog)**.**

$\forall w, \varphi, p, \vec{q}$, φ *is* $\Box$ *free, then*

(1) φ @ $(w, q :\!- \vec{p}), \vec{p} \to q$ @ $w \vdash_\Sigma \varphi$ @ w
(2) φ @ $w, \vec{p} \to q$ @ $w \vdash_\Sigma \varphi$ @ $(w, q :\!- \vec{p})$

Proof (sketch): By induction over the structure of φ. We prove (1) and (2) simultaneously. When φ is of the form $\top$ or $\bot$, the conclusion follows directly from the $\top$R and $\bot$L rules, respectively. When φ is of the form $S \in$ **At**, (1) follows from the Dlog and $\to$R rules, and (2) follows from Conv. The proofs for the rest of the cases where φ involves some connective follow a similar pattern. The conclusion follows from applying the left and right rules for the connective on the result(s) of applying the induction hypothesis on subformulas of φ. □

Lemma 13 (Axiom MP). *If $\mathcal{E}_1 :: \cdot \vdash \varphi_1$ @ w and $\mathcal{E}_2 :: \cdot \vdash \varphi_1 \to \varphi_2$ @ w then $\exists \mathcal{E} :: \cdot \vdash \varphi_2$ @ w*

Proof (sketch): $\mathcal{D} :: \varphi_1$ @ $w, \varphi_1 \to \varphi_2$ @ $w \vdash_\Sigma \varphi_2$ @ w may be constructed by applying $\to L$ and weakening on derivations from Init. The conclusion follows by cutting φ_1 @ w and $\varphi_1 \to \varphi_2$ @ w from $\mathcal{D}$. □

Lemma 28 (Generalized Axiom N)**.**

If $\mathcal{E} :: \Gamma \vdash_\Sigma \Delta$, then $\exists \mathcal{D} :: \Gamma$ @ $w \vdash_\Sigma \Delta$ @ w
Where we define Γ @ w as follows.

$$\begin{array}{lcl} \cdot \text{ @ } w & = & \cdot \\ (\Gamma, \varphi \text{ @ } w') \text{ @ } w & = & \Gamma \text{ @ } w, \varphi \text{ @ } (w, w') \end{array}$$

Proof (sketch): By induction over the structure of $\mathcal{E}$. For the cases where $\mathcal{E}$ ends in Init, $\top$ R, or $\bot$ L, the conclusion follows directly from the Init, $\top$ R, and $\bot$ L rules, respectively. Most of the rest of the cases proceed the same way: the conclusion follows from applying the same rule on the result of applying the induction hypothesis on the subderivations of $\mathcal{E}$. The only exception is when $\mathcal{E}$ ends in Conv, where an additional derivation comes from applying $\wedge$ L1, $\wedge$ L2, and weakening on a derivation from Init. □

The proof of Axiom Mon follows from a more general lemma, which requires a few additional lemmas to prove.

Lemma 15 (Axiom Mon.) *If $\vdash_\Sigma \gamma_1 \to \gamma_2$ @ $\top$ then $\forall w$ $\vdash_\Sigma \Box_{\gamma_2} \varphi^+ \to \Box_{\gamma_1} \varphi^+$ @ w.*

Proof (sketch): $\mathcal{D} :: \gamma_1 \wedge (\wedge w)$ @ $\top \vdash (\gamma_2 \wedge (\wedge w))$ @ $\top$ may be obtained by applying $\wedge R$ on derivations obtained from Lemma 30 and application of Init, weakening, and $\wedge R$. The conlusion follows from applying weakening, $\Box$R, $\Box$L, and $\to$R on the result of applying Lemma 31 on $\mathcal{D}$. □

Lemma 29.

If $\mathcal{E} :: \Gamma, \gamma_1$ @ $w \vdash_\Sigma \gamma_2$ @ $w, \Delta, \Delta_\to$ where $\forall J \in \Delta_\to$, $J = \gamma_1 \to \gamma_2$ @ w, then $\forall w, \Gamma, \gamma_1$ @ $w \vdash_\Sigma \gamma_2$ @ w, Δ.

Proof (sketch): By induction over the structure of $\mathcal{E}$. □

Lemma 30. *If $\mathcal{E} ::\vdash \gamma_1 \to \gamma_2$ @ $\top$ then γ_1 @ $\top \vdash \gamma_2$ @ $\top$.*

Proof (sketch): By weakening, we have γ_1 @ $\top \vdash \gamma_2$ @ $\top, \gamma_1 \to \gamma_2$ @ $\top$. Then, we can apply Lemma 29, with $\Gamma = \Delta = \cdot$ to reach the conclusion. □

Lemma 31 (Generalized Axiom Mon)**.**

If γ_1 @ $\top \vdash_\Sigma \gamma_2$ @ $\top$ then $\forall w$,

(1) φ^+ @ $(w, \gamma_2) \vdash_\Sigma \varphi^+$ @ (w, γ_1) and
(2) φ^- @ $(w, \gamma_1) \vdash_\Sigma \varphi^-$ @ (w, γ_2)

Proof (sketch): By induction over the structure of φ. When $\varphi = p$, e can show that $\gamma_1 \wedge (\wedge w)$ @ $\top \vdash (\gamma_2 \wedge (\wedge w))$ @ $\top$. Then we can use the Conv rule to reach the conclusion. When $\varphi^+ = \Box_\gamma \varphi_1^+$ the proof proceeds as follows: Given any w', we can apply I.H. on φ_1^+ and instantiate w with $(w', \overline{\gamma})$ to obtain φ_1^+ @ $(w', \overline{\gamma}, \overline{\gamma_2}) \vdash \varphi_1^+$ @ $(w', \overline{\gamma}, \overline{\gamma_1})$. Then we can apply $\Box$R and $\Box$L rules to reach the conclusion. □

It is not readily aparent that Axiom Mon follows from Lemma 15. Becker et al. [5] explains that Axiom Mon says that submitting more or strengthening policies in the counterfactual statement $\Box_\gamma \varphi$ should not make the formula φ false. This is our definition of positive formulas from Section 4, so Lemma 15 is, in fact, a more general statement which entails Axiom Mon.