Technical Report
The Deduction System of Matching Logic

Formal Systems Laboratory[1]

[1]University of Illinois, Urbana-Champaign, USA

April 20, 2017

Abstract

Abstract goes here.

1 Syntax

Formulas of matching logic, called *patterns*, are written in a formal language, denoted as $\mathcal{L}$, whose grammar is listed in (1). The language $\mathcal{L}$ is many-sorted. A signature of $\mathcal{L}$ contains not only a finite set Σ of *symbols*, but also a finite nonempyty set S of *sorts*. Each symbol $\sigma \in \Sigma$ is, of course, sorted, with a fixed nonempty arity. We write $\sigma \in \Sigma_{s_1,\ldots,s_n,s}$ to emphasize that σ takes n arguments (with argument sorts $s1, \ldots, s_n$) and returns a pattern in sort s, but we hope in most cases sorting is clear from context.

The grammar for $\mathcal{L}$, as defined below, is almost identical to first-order logic, except that in $\mathcal{L}$ there is no difference between relational (predicate) and functional symbols, and we accept first-order terms as patterns in matching logic.

$$\begin{aligned} P ::= {} & x \\ & \mid P_1 \wedge P_2 \\ & \mid \neg P \\ & \mid \forall x.P \\ & \mid \sigma(P_1, \ldots, P_n), \end{aligned} \tag{1}$$

where the universal quantifier ($\forall x$) behaves the same as in first-order logic, with alpha-renaming always assumed.

For simplicity, we did not mention sorting in the grammar definition, and assume it should be clear to all readers. For example, in $P_1 \wedge P_2$, both patterns P_1 and P_2 should have the same sort, and that sort is the sort of $P_1 \wedge P_2$. The sort of $\forall x.P$ is the sort of P, while the sort of variable x does not matter. To see why it is the case, consider the pattern $\exists x.list(x, 1 \cdot 3 \cdot 5)$, which is the set of all memory configurations that has a list $(1, 3, 5)$ in it.

Propositional connectives are always assumed, including disjunction ($\vee$), implication ($\rightarrow$), and equivalence ($\leftrightarrow$). Existential quantifier ($\exists x$) is defined by universal quantifier ($\forall x$) in the normal way. The bottom pattern ($\bot_s$) and the top pattern ($\top_s$) in sort s are given by $x \wedge \neg x$ and $\neg \bot_s$, respectively, where x is a variable in sort s. It does not matter which variable we pick.

1.1 Extended syntax

The formal language $\mathcal{L}$ is often extended with *definedness* symbols. For s_1, s_2 are two sorts, the definedness symbol $\lceil \_ \rceil^{s_2}_{s_1} \in \Sigma_{s_1,s_2}$ is a unary symbol with one argument sort s_1 and the result sort s_2. For a pattern P who has sort s_1, the pattern $\lceil \_ \rceil^{s_2}_{s_1}(P)$ is often written as $\lceil P \rceil^{s_2}_{s_1}$, or simply $\lceil P \rceil$.

Definedness symbols carry specific intended semantics. For each definedness symbol $\lceil \_ \rceil^{s_2}_{s_1}$, we add the pattern $\lceil x \rceil^{s_2}_{s_1}$ as an axiom to the deductive system, where x is a variable who has sort s_1. It does not matter which variable we pick.

With definedness symbols, we extends the formal language $\mathcal{L}$ with

$$
\begin{aligned}
&\lfloor P \rfloor^{s_2}_{s_1} := \neg \lceil \neg P \rceil^{s_2}_{s_1} \\
&P_1 =^{s_2}_{s_1} P_2 := \lfloor P_1 \leftrightarrow P_2 \rfloor^{s_2}_{s_1} \\
&P_1 \neq^{s_2}_{s_1} P_2 := \neg (P_1 =^{s_2}_{s_1} P_2) \\
&P_1 \subseteq^{s_2}_{s_1} P_2 := \lfloor P_1 \rightarrow P_2 \rfloor^{s_2}_{s_1} \\
&x \in^{s_2}_{s_1} P := x \subseteq^{s_2}_{s_1} P.
\end{aligned}
$$

Remark 1. *To prevent writing tangled subscripts and superscripts that indicate sorts of variables and patterns all the time, we omit them as much as possible, unless there is a chance of confusing things. A statement with sorting subscripts and superscripts omitted is treated as (possibly many) statements with the omitting sorting subscripts and superscripts completed in all possible well-formed ways.*

2 Deductive System

A deductive system is a recursive set of patterns as *axioms* and a finite set of *inference rules*. The deductive system of matching logic that we introduce in this section has been proved *sound* and *complete*.

2.1 The deductive system

Axioms are given by the following axiom schemata where P, Q, R are arbitrary patterns and x, y are logic variables.

- (K1) $P \rightarrow (Q \rightarrow P)$
- (K2) $(P \rightarrow (Q \rightarrow R)) \rightarrow ((P \rightarrow Q) \rightarrow (P \rightarrow R))$
- (K3) $(\neg P \rightarrow \neg Q) \rightarrow (Q \rightarrow P)$

- (K4) $\forall x.P \to P[y/x]$
- (K5) $\forall x.(P \to Q) \to (P \to \forall x.Q)$ if x does not occur free in P
- (K6) $P_1 = P_2 \to (Q[P_1/x] \to Q[P_2/x])$
- (Df) $\lceil x \rceil$
- (M1) $x \in y = (x = y)$
- (M2) $x \in P \wedge Q = (x \in P) \wedge (x \in Q)$
- (M3) $x \in \neg P = \neg(x \in P)$
- (M4) $x \in \forall y.P = \forall y.x \in P$ where x is distinct from y
- (M5) $x \in \sigma(\dots, P_i, \dots) = \exists y.y \in P_i \wedge x \in \sigma(\dots, y, \dots)$ where y occurs free in the left hand side of the equation.

Remark 2. *Substitution is denoted as $Q[P/x]$. Alpha-renaming is always assumed in order to avoid free variables capturing.*

Inference rules include

- (Modus Ponens) From P and $P \to Q$, deduce Q.
- (Universal Generalization) From P, deduce $\forall x.P$.
- (Membership Introduction) From P, deduce $\forall x.(x \in P)$, where x does not occur free in P.
- (Membership Elimination) From $\forall x.(x \in P)$, deduce P, where x does not occur free in P.

Theorem 3. *The proof system is sound and complete.*

Proof. No proof. □

2.2 Metatheorems of the deductive system

Writing formal proofs is never easy. Derivations are prone to be lengthy and boring. To ease such difficulty, we here in this section introduce a dozen of lemmas (i.e., metatheorems) of the deductive system. Metatheorems discover all kinds of properties of the deductive system, from the simplest "$\vdash P = P$" to the complexest deduction theorem and the framing rule.

Proposition 4 (Tautology)**.** *For any propositional tautology $\mathcal{A}(p_1, \dots, p_n)$ where $p_1, \dots, p_n$ are all propositional variables in $\mathcal{A}$, and for any patterns $P_1, \dots, P_n$,*

$$\vdash \mathcal{A}(P_1, \dots, P_n).$$

Proof. No proof. □

Corollary 5. $\vdash \top$.

Proof. By definition, $\top = \neg\bot = \neg(x \wedge \neg x)$, where x is a matching logic variable who has the same sort with $\top$. Let proposition $\mathcal{A} = \neg(p \wedge \neg p)$ with p is a propositional variable. Then $\mathcal{A}$ is a propositional tautology. By Proposition 4, $\top = \mathcal{A}[x/p]$ is derivable in the proof system, i.e., $\vdash \top$. □

Proposition 6 ($\vee$-Introduction)**.** $\vdash P$ *implies* $\vdash P \vee Q$.

Proof.

$$\dfrac{\dfrac{\dfrac{\cdot}{\vdash P}\quad\dfrac{\cdot}{\vdash P \to (\neg Q \to P)}\ \text{(K1)}}{\vdash \neg Q \to P}\ \text{(MP)}}{\vdash P \vee Q}\ \text{(Sugar)}$$

□

Corollary 7 ($\to$-Introduction)**.** $\vdash P$ *implies* $\vdash Q \to P$ *and* $\vdash \neg P \to Q$.

Remark 8. *In general,* $\vdash P \vee Q$ *does not implies* $\vdash P$ *or* $\vdash Q$*. For example, we have shown that* $\vdash \top$*, and* $\top$ *is, by definition, just sugar of* $\neg\bot = \neg(x \wedge \neg x) = \neg x \vee x$*. It is clearly wrong if we conclude* $\vdash \neg x$ *or* $\vdash x$*. From a semantic point of view, it is easy to understand: the union of two sets is the total set does not imply that one of them is the total set.*

Proposition 9 ($\wedge$-Introduction and Elimination)**.** $\vdash P$ *and* $\vdash Q$ *iff* $\vdash P \wedge Q$.

Proof. ($\Rightarrow$).

$$\dfrac{\dfrac{\cdot}{\vdash P}\quad\dfrac{\dfrac{\cdot}{\vdash Q}\quad\dfrac{\cdot}{\vdash Q \to P \to P \wedge Q}\ \text{(Taut)}}{\vdash P \to P \wedge Q}\ \text{(MP)}}{\vdash P \wedge Q}\ \text{(MP)}$$

($\Leftarrow$). Left as an exercise. □

Equalities plays an important role in matching logic. Axiom (K6) is very powerful even though it looks quite simple. It basically says that whenever one establishes that $P = Q$, then the two patterns are interchangeable everywhere in any patterns, as concluded in the next lemma.

Lemma 10. *If* $\vdash P_1 = P_2$ *and* $\vdash Q[P_1/x]$*, then* $\vdash Q[P_2/x]$.

Proof.

$$\dfrac{\dfrac{\dfrac{\cdot}{\vdash P_1 = P_2}\quad\dfrac{\cdot}{\vdash P_1 = P_2 \to (Q[P_1/x] \to Q[P_2/x])}\ \text{(K6)}}{\vdash Q[P_1/x] \to Q[P_2/x]}\ \text{(MP)}\quad\dfrac{\cdot}{\vdash Q[P_1/x]}}{\vdash Q[P_2/x]}\ \text{(MP)}$$

□

Remind that the equality "=" is not a built-in logic connective in matching logic. It is the syntactic sugar of $\neg\lceil\neg(P \leftrightarrow Q)\rceil$, where $\lceil\_\rceil$ is the definedness symbol that we introduced before. One may wonder how to establish such equalities in the deduction system. Indeed, the proof system says little about how to derive an equality. Most of its axioms (except (K6)) and rules are not even about equalities. That is why the next proposition is quite useful in practice. It helps one to establish an equality pattern.

Proposition 11. *$\vdash P \leftrightarrow Q$ iff $\vdash P = Q$.*

Proof. That the right hand side implies the left is easy, so we left the proof as an exercise to the readers. In the following, we only prove that the left implies the right. By definition, $P = Q$ is the syntactic sugar of $\neg\lceil\neg\,(P \leftrightarrow Q)\rceil$, so we have the following derivation.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\vdots}{\vdash P \leftrightarrow Q}}{\vdash \forall y.(y \in P \leftrightarrow Q)}\,(\in\text{-Intro})}{\vdash y \in P \leftrightarrow Q}\,(\text{K4, MP})}{\vdash \neg(x \in \lceil y\rceil) \vee (y \in P \leftrightarrow Q)}\,(\vee\text{-Intro})}{\vdash \forall y.(\neg(x \in \lceil y\rceil) \vee (y \in P \leftrightarrow Q)}\,(\forall y\text{-Gen})}{\vdash \neg\exists y.\,(x \in \lceil y\rceil \wedge y \in \neg\,(P \leftrightarrow Q))}\,(\text{Sugar, K3, M3})}{\vdash \neg\,(x \in \lceil\neg\,(P \leftrightarrow Q)\rceil)}\,(\text{K6, M5})}{\vdash x \in \neg\lceil\neg\,(P \leftrightarrow Q)\rceil}\,(\text{K6, M3})}{\vdash \forall x.\,(x \in \neg\lceil\neg\,(P \leftrightarrow Q)\rceil)}\,(\forall x\text{-Gen})}{\vdash \neg\lceil\neg\,(P \leftrightarrow Q)\rceil}\,(\in\text{-Intro})\quad.$$

□

Remark 12. *We never say $P = Q$ and $P \leftrightarrow Q$ are logically equivalent. One will never derive $\vdash (P = Q) = (P \leftrightarrow Q)$ for any patterns P and Q in the deductive system from a consistent set of axioms.*

Corollary 13. *The following propositions hold for any pattern P.*

1. $\vdash P$ *iff* $\vdash P = \top$.
2. $\vdash \neg P$ *iff* $\vdash P = \bot$.
3. $\vdash (P \wedge \top) = P$.
4. $\vdash (P \wedge \bot) = \bot$.
5. $\vdash (P \vee \top) = \top$.
6. $\vdash (P \vee \bot) = P$.
7. $\vdash \forall x.\top = \top$.
8. $\vdash \forall x.\bot = \bot$.
9. $\vdash \exists x.\top = \top$.

10. $\vdash \exists x.\bot = \bot$.

11. $\vdash (x \in \top) = \top$.

12. $\vdash (x \in \bot) = \bot$.

13. $\vdash P = P$.

Proposition 14 (Functional Substitution)**.** $\vdash \exists y.(Q = y) \to (\forall x.P \to P[Q/x]))$*, if* y *occurs free in* Q*.*

Proposition 15. $\vdash x \in \lceil y \rceil$.

Proof.

$$\begin{aligned} &\vdash x \in \lceil y \rceil \\ \text{if } &\vdash \forall x.(x \in \lceil y \rceil) && (K5,\ K6, \text{ and Modus Ponens}) \\ \text{iff } &\vdash \lceil y \rceil. \end{aligned}$$

□

Proposition 16. $\vdash P \to \lceil P \rceil$.

Proof.

$$\begin{aligned} &\vdash P \to \lceil P \rceil \\ \text{iff } &\vdash \forall x.(x \in P \to \lceil P \rceil) \\ \text{if } &\vdash x \in P \to \lceil P \rceil \\ \text{iff } &\vdash x \in P \to x \in \lceil P \rceil \\ \text{iff } &\vdash x \in P \to \exists y.(y \in P \wedge x \in \lceil y \rceil) \\ \text{iff } &\vdash x \in P \to \neg\forall y.(y \notin P \vee x \notin \lceil y \rceil) \\ \text{iff } &\vdash \forall y.(y \notin P \vee x \notin \lceil y \rceil) \to x \notin P \\ \text{if } &\vdash x \notin P \vee x \notin \lceil x \rceil \to x \notin P \\ \text{iff } &\vdash x \in P \to x \in P \wedge x \in \lceil x \rceil \\ \text{iff } &\vdash x \in P \to x \in \lceil x \rceil \\ \text{if } &\vdash x \in \lceil x \rceil \end{aligned}$$

Remark Similarly we can show $\vdash \lfloor P \rfloor \to P$. □

Proposition 17. $\vdash \forall x.(x \in P) = \lfloor P \rfloor$*, where* x *occurs free in* P*.*

Proof. By Proposition 11 and 9, it suffices to show

$$\vdash \forall x.(x \in P) \to \lfloor P \rfloor \tag{2}$$

and

$$\vdash \lfloor P \rfloor \to \forall x.(x \in P). \tag{3}$$

To show (2),

$$
\begin{aligned}
&\vdash \forall x.(x \in P) \to \lfloor P \rfloor \\
\text{iff } &\vdash \forall x.\lceil x \wedge P \rceil \to \neg\lceil \neg P \rceil \\
\text{iff } &\vdash \lceil \neg P \rceil \to \exists x.\neg\lceil x \wedge P \rceil \\
\text{iff } &\vdash \forall y.(y \in (\lceil \neg P \rceil \to \exists x.\neg\lceil x \wedge P \rceil)) \\
\text{if } &\vdash y \in (\lceil \neg P \rceil \to \exists x.\neg\lceil x \wedge P \rceil \\
\text{iff } &\vdash \exists z_1.(z_1 \notin P \wedge y \in \lceil z_1 \rceil) \to \\
&\qquad \exists x.\neg(\exists z_2.(z_2 = x \wedge z_2 \in P \wedge y \in \lceil z_2 \rceil)) \\
\text{iff } &\vdash \exists z_1.(z_1 \notin P \wedge \top) \to \qquad \text{(Proposition 15, ??, and Corollary ??)} \\
&\qquad \exists x.\neg(\exists z_2.(z_2 = x \wedge z_2 \in P \wedge \top)) \\
\text{iff } &\vdash \exists z_1.(z_1 \notin P) \to \exists x.\neg(\exists z_2.(z_2 = x \wedge z_2 \in P)) \\
\text{iff } &\vdash \forall x.(\exists z_2.(z_2 = x \wedge z_2 \in P)) \to \forall z_1.(z_1 \in P) \\
\text{if } &\vdash \forall z_1.(\forall x.(\exists z_2.(z_2 = x \wedge z_2 \in P)) \to (z_1 \in P)) \\
\text{if } &\vdash \forall x.(\exists z_2.(z_2 = x \wedge z_2 \in P)) \to (z_1 \in P).
\end{aligned}
$$

Since $\vdash \forall x.(\exists z_2.(z_2 = x \wedge z_2 \in P)) \to \exists z_2.(z_2 = z_1 \wedge z_2 \in P)$, it suffices to show

$$
\begin{aligned}
&\vdash \exists z_2.(z_2 = z_1 \wedge z_2 \in P) \to (z_1 \in P) \\
\text{iff } &\vdash z_1 \notin P \to \forall z_2.(z_2 \neq z_1 \vee z_2 \notin P) \\
\text{if } &\vdash \forall z_2.(z_1 \notin P \to z_2 \neq z_1 \vee z_2 \notin P) \\
\text{if } &\vdash z_1 \notin P \to z_2 \neq z_1 \vee z_2 \notin P \\
\text{if } &\vdash z_2 = z_1 \wedge z_2 \in P \to z_1 \in P.
\end{aligned}
$$

And we proved (2).

Similarly, to show (3),

$$
\begin{aligned}
&\vdash \lfloor P \rfloor \to \forall x.(x \in P) \\
\text{iff } &\vdash \exists x.\neg\lceil x \wedge P \rceil \to \lceil \neg P \rceil \\
\text{iff } &\vdash \forall y.(y \in \exists x.\neg\lceil x \wedge P \rceil \to \lceil \neg P \rceil) \\
\text{if } &\vdash y \in \exists x.\neg\lceil x \wedge P \rceil \to \lceil \neg P \rceil \\
\text{iff } &\vdash \exists x.\neg\exists z_2.(z_2 = x \wedge z_2 \in P) \to \exists z_1.(z_1 \notin P) \\
\text{iff } &\vdash \forall z_1.(z_1 \in P) \to \exists z_2.(z_2 = x \wedge z_2 \in P) \\
\text{iff } &\vdash x \in P \to \exists z_2.(z_2 = x \wedge z_2 \in P).
\end{aligned}
$$

We proved (3).

Remark If x occurs free in P, the result does not hold. For example, let P be $upto(x)$ where $upto(\cdot)$ is interpreted to $upto(n) = \{0, 1, \ldots, n\}$ on $\mathbb{N}$. □

Remark From Membership Introduction and Elimination inference rules and Proposition 17, $\vdash P$ iff $\vdash \lfloor P \rfloor$.

Proposition 18 (Classification Reasoning)**.** *For any P and Q, from $\vdash P \to Q$ and $\vdash \neg P \to Q$ deduce $\vdash Q$.*

Proof. From $\vdash \neg P \to Q$ deduce $\vdash \neg Q \to P$. Notice that $\vdash P \to Q$, so we have $\vdash \neg Q \to Q$, i.e., $\vdash \neg\neg Q \vee Q$ which concludes the proof. □

Corollary 19. *For any P_1, P_2, and Q are patterns with $\vdash P_1 \vee P_2$, from $\vdash P_1 \to Q$ and $\vdash P_2 \to Q$, deduce $\vdash Q$.*

Definition 20 (Predicate Pattern)**.** *A pattern P is called a predicate pattern or a predicate if $\vdash (P = \top) \vee (P = \bot)$.*

Remark Predicate patterns are closed under all logic connectives.

Remark For any P, $\lceil P \rceil$ is a predicate pattern.

Proposition 21. $\vdash (\lceil P \rceil = \bot) = (P = \bot)$ *and* $\vdash (\lfloor P \rfloor = \top) = (P = \top)$.

Proof. It is easy to prove one derivation from the other, so we only prove the first one. By Proposition 11, it suffices to prove

$$\vdash (\lceil P \rceil = \bot) \to (P = \bot) \tag{4}$$

and

$$\vdash (P = \bot) \to (\lceil P \rceil = \bot) \tag{5}$$

The proof of (5) is trivial and we left it as an exercise. We now prove (4) through the following backward reasoning.

$$\begin{aligned}
& \vdash (\lceil P \rceil = \bot) \to (P = \bot) \\
\text{iff} \quad & \vdash \forall y.(y \in ((\lceil P \rceil = \bot) \to (P = \bot))) \\
\text{if} \quad & \vdash y \in ((\lceil P \rceil = \bot) \to (P = \bot)) \\
\text{iff} \quad & \vdash (y \in (\lceil P \rceil = \bot) \to (y \in (P = \bot)).
\end{aligned} \tag{6}$$

While for any pattern Q,

$$\begin{aligned}
& \vdash y \in (Q = \bot) \\
\text{iff} \quad & \vdash y \in \neg\lceil \neg(Q \leftrightarrow \bot) \rceil \\
\text{iff} \quad & \vdash y \in \neg\lceil Q \rceil \\
\text{iff} \quad & \vdash \neg\exists z.(z \in Q \wedge y \in \lceil z \rceil) \\
\text{iff} \quad & \vdash \neg\exists z.(z \in Q)
\end{aligned}$$

So we continue to prove (6) by showing

$$\begin{array}{ll} & \vdash (y \in (\lceil P \rceil = \bot)) \to (y \in (P = \bot)) \\ \text{iff} & \vdash \neg\exists z.(z \in \lceil P \rceil) \to \neg\exists z.(z \in P) \\ \text{iff} & \vdash \exists z.(z \in P) \to \exists z.(z \in \lceil P \rceil) \\ \text{iff} & \vdash \exists z.(z \in P) \to \exists z.(\exists z_1.(z_1 \in P \wedge z \in \lceil z_1 \rceil)) \\ \text{iff} & \vdash \exists z.(z \in P) \to \exists z.\exists z_1.(z_1 \in P) \\ \text{iff} & \vdash \exists z_1.(z_1 \in P) \to \exists z.\exists z_1.(z_1 \in P). \end{array}$$

And we finish the proof by noticing the fact that for any pattern Q and variable x,

$$\vdash Q \to \exists x.Q.$$

□

Proposition 22. *For any predicate P, $\vdash (P \neq \top) = (P = \bot)$ and $\vdash (P \neq \bot) = (P = \top)$.*

Proof. We only prove the first derivation, by showing both

$$\vdash (P \neq \top) \to (P = \bot) \tag{7}$$

and

$$\vdash (P = \bot) \to (P \neq \top). \tag{8}$$

Proving (8) is trivial. We now prove (7), which is also trivial by transforming disjunction to implication. □

Proposition 23. *For any pattern Q and any predicate pattern P, $\vdash P \vee Q$ iff $\vdash P \vee \lfloor Q \rfloor$.*

Proof. ($\Leftarrow$) is obtained immediately by the remark of Proposition 16. We now prove ($\Rightarrow$).

Because $\vdash Q = \top \vee Q \neq \top$, it suffices to show

$$\vdash Q = \top \to (P \vee \lfloor Q \rfloor = \top) \tag{9}$$

and

$$\vdash Q \neq \top \to (P \vee \lfloor Q \rfloor = \top) \tag{10}$$

by Corollary 19, and the fact that $\vdash P \vee \lfloor Q \rfloor = \top$ and $\vdash \top$ imply $\vdash P \vee \lfloor Q \rfloor$.

The proof of (9) is straightforward as follows.

$$\begin{array}{ll} & \vdash Q = \top \to (P \vee \lfloor Q \rfloor = \top) \\ \text{if} & \vdash Q = \top \to (P \vee \lfloor \top \rfloor = \top) \\ \text{if} & \vdash Q = \top \to (\top = \top) \\ \text{if} & \vdash \top. \end{array}$$

The proof of (10) needs more effort:

$$\begin{aligned}
& \vdash Q \neq \top \to (P \vee \lfloor Q \rfloor = \top) \\
\text{iff} \quad & \vdash (Q = \top) \vee (P \vee \lfloor Q \rfloor = \top) \\
\text{iff} \quad & \vdash (\lfloor Q \rfloor = \top) \vee (P \vee \lfloor Q \rfloor = \top) \\
\text{iff} \quad & \vdash \lfloor Q \rfloor \neq \top \to (P \vee \lfloor Q \rfloor = \top) \\
\text{iff} \quad & \vdash \lfloor Q \rfloor = \bot \to (P \vee \lfloor Q \rfloor = \top) \\
\text{if} \quad & \vdash \lfloor Q \rfloor = \bot \to (P \vee \bot = \top) \\
\text{iff} \quad & \vdash \lfloor Q \rfloor = \bot \to (P = \top) \\
\text{if} \quad & \vdash Q = \top \vee P = \top.
\end{aligned}$$

Notice that P is a predicate pattern, so it suffices to show

$$\vdash P = \top \to (Q = \top \vee P = \top),$$

whose validity is obvious, and

$$\vdash P = \bot \to (Q = \top \vee P = \top),$$

which is proved by showing

$$\vdash P = \bot \to Q = \top. \tag{11}$$

Because $\vdash P \vee Q$, it suffices to show

$$\begin{aligned}
& \vdash P = \bot \to (P \vee Q) \to (Q = \top) \\
\text{if} \quad & \vdash P = \bot \to (\bot \vee Q) \to (Q = \top) \\
\text{iff} \quad & \vdash P = \bot \to Q \to (Q = \top) \\
\text{if} \quad & \vdash Q \to (Q = \top) \\
\text{iff} \quad & \vdash (Q \neq \top) \to \neg Q \\
\text{iff} \quad & \vdash (\lfloor Q \rfloor = \bot) \to \neg Q.
\end{aligned}$$

Notice we have $\vdash Q \to \lfloor Q \rfloor$, which means $\vdash \neg\lfloor Q \rfloor \to \neg Q$, so it suffices to show

$$\begin{aligned}
& \vdash (\lfloor Q \rfloor = \bot) \to \neg\lfloor Q \rfloor \\
\text{iff} \quad & \vdash (\lfloor Q \rfloor = \bot) \to \neg\bot \\
\text{iff} \quad & \vdash (\lfloor Q \rfloor = \bot) \to \top \\
\text{iff} \quad & \vdash \top.
\end{aligned}$$

And this concludes the proof. □

Proposition 24 (Deduction Theorem)**.** *If $\Gamma \cup \{P\} \vdash Q$ and the derivation does not use $\forall x$-Generalization where x is free in P, then $\Gamma \vdash \lfloor P \rfloor \to Q$.*

Proof. The proof is by induction on n, the length of the derivation of Q from $\Gamma \cup \{P\}$.

Base step: $n = 1$, and Q is an axiom, or P, or a member of Γ. If Q is an axiom or a member of Γ, then $\Gamma \vdash Q$ and as a result, $\Gamma \vdash \lfloor P \rfloor \rightarrow Q$. If Q is P, then $\Gamma \vdash \lfloor P \rfloor \rightarrow Q$ by Proposition 16.

Induction step: Let $n > 1$. Suppose that if P' can be deduced from $\Gamma \cup \{P\}$ without using $\forall x$-Generalization where x is free in P, in a derivation containing fewer than n steps, then $\Gamma \vdash \lfloor P \rfloor \rightarrow P'$.

Case 1: Q is an axiom, or P, or a member of Γ. Precisely as in the Base step, we show that $\vdash \lfloor P \rfloor \rightarrow Q$.

Case 2: Q follows from two previous patterns in the derivation by an application of Modus Ponens. These two patterns must have the forms Q_1 and $Q_1 \rightarrow Q$, and each one can certainly be deduced from $\Gamma \cup \{P\}$ by a derivation with fewer than n steps, by just omitting the subsequent members from the original derivation from $\Gamma \cup \{P\} \vdash Q$. So we have $\Gamma \cup \{P\} \vdash Q_1$ and $\Gamma \cup \{P\} \vdash Q_1 \rightarrow Q$, and, applying the hypothesis of induction, $\Gamma \vdash \lfloor P \rfloor \rightarrow Q_1$ and $\Gamma \vdash \lfloor P \rfloor \rightarrow (Q_1 \rightarrow Q)$. It follows immediately that $\Gamma \vdash \lfloor P \rfloor \rightarrow Q$.

Case 3: Q follows from a previous pattern in the derivation by an application of $\forall x_i$-Generalization where x_i does not occur free in P. So Q is $\forall x_i.Q_1$, say, and Q_1 appears previously in the derivation. Thus $\Gamma \cup \{P\} \vdash Q_1$, and the derivation has fewer than n steps, so $\Gamma \vdash \lfloor P \rfloor \rightarrow Q_1$, since there is no application of Universal Generalization involving a free variable of P. Also x_i cannot occur free in P, as it is involved in an application of Universal Generalization in the deduction of Q from $\Gamma \cup \{P\}$. So we have a derivation of $\Gamma \vdash \lfloor P \rfloor \rightarrow Q$ as follows.

$$\begin{aligned} & \Gamma \vdash \lfloor P \rfloor \rightarrow Q \\ \text{iff} \quad & \Gamma \vdash \lfloor P \rfloor \rightarrow \forall x_i.Q_1 \\ \text{if} \quad & \Gamma \vdash \forall x_i.(\lfloor P \rfloor \rightarrow Q_1) \\ \text{if} \quad & \Gamma \vdash \lfloor P \rfloor \rightarrow Q_1. \end{aligned}$$

So $\Gamma \vdash \lfloor P \rfloor \rightarrow Q$ as required.

Case 4: Q follows from a previous pattern in the derivation by an application of Membership Introduction. So Q is $\forall x_i.(x_i \in Q_1)$ with x_i is free in Q_1, say, and Q_1 appears previously in the derivation. Thus $\Gamma \cup \{P\} \vdash Q_1$, and the derivation has fewer than n steps, so $\Gamma \vdash \lfloor P \rfloor \rightarrow Q_1$, since there is no application of Universal Generalization involving a free variable of P. So we have a derivation of $\Gamma \vdash \lfloor P \rfloor \rightarrow Q$ as follows.

$$\begin{aligned} & \Gamma \vdash \lfloor P \rfloor \rightarrow Q \\ \text{iff} \quad & \Gamma \vdash \lfloor P \rfloor \rightarrow \forall x_i.(x_i \in Q_1) \\ \text{iff} \quad & \Gamma \vdash \lfloor P \rfloor \rightarrow \lfloor Q_1 \rfloor, \end{aligned}$$

which follows by the hypothesis of induction $\Gamma \vdash \lfloor P \rfloor \rightarrow Q_1$ and the fact that $\Gamma \vdash Q_1 \rightarrow \lfloor Q_1 \rfloor$ (by the Remark in Proposition 16).

Case 5: Q follows from a previous pattern in the derivation by an application of Membership Elimination. The previous pattern must have the form $\forall x_i.(x_i \in Q)$, and can be deduced from $\Gamma \cup \{P\}$ by a derivation with fewer than n steps, by just omitting

the subsequent members from the original derivation from $\Gamma \cup \{P\} \vdash Q$. So we have $\Gamma \cup \{P\} \vdash \forall x_i.(x_i \in Q)$, and, applying the hypothesis of induction, $\Gamma \vdash \lfloor P \rfloor \to \forall x_i.(x_i \in Q)$. So we have a derivation of $\Gamma \vdash \lfloor P \rfloor \to Q$ as follows.

$$\begin{array}{rll} & \Gamma \vdash \lfloor P \rfloor \to Q & \\ \text{iff} & \Gamma \vdash \neg\lfloor P \rfloor \vee Q & \\ \text{iff} & \Gamma \vdash \neg\lfloor P \rfloor \vee \lfloor Q \rfloor & \text{(Proposition 23)} \\ \text{iff} & \Gamma \vdash \neg\lfloor P \rfloor \vee \forall x_i.(x_i \in Q) & \\ \text{iff} & \Gamma \vdash \lfloor P \rfloor \to \forall x_i.(x_i \in Q), & \end{array}$$

which is the hypothesis of induction. And this concludes our inductive proof. □

Corollary 25 (Closed-form Deduction Theorem)**.** *If P is closed, $\Gamma \cup \{P\} \vdash Q$ implies $\Gamma \vdash \lfloor P \rfloor \to Q$.*

Theorem 26 (Frame Rule)**.** *Let $\sigma \in \Sigma$ be a symbol in the signature. From $P_1 \to P_2$, deduce $\sigma(P_1) \to \sigma(P_2)$. In its most general form, $P_1 \to P_2$ deduces $\sigma(Q_1, \ldots, P_1, \ldots, Q_n) \to \sigma(Q_1, \ldots, P_2, \ldots, Q_n)$.*

Proof. we write $\sigma(Q_1, \ldots, P_i, \ldots, Q_n)$ as $\sigma(P_i, \vec{Q})$ for short, for any $i \in \{1, 2\}$.

$$\begin{array}{rl} & \vdash \sigma(P_1, \vec{Q}) \to \sigma(P_2, \vec{Q}) \\ \text{iff} & \vdash y \in (\sigma(P_1, \vec{Q}) \to \sigma(P_2, \vec{Q})) \\ \text{iff} & \vdash (y \in \sigma(P_1, \vec{Q})) \to (y \in \sigma(P_2, \vec{Q})) \\ \text{iff} & \vdash \exists z_1.\exists \vec{z}.(z_1 \in P_1 \wedge \vec{z} \in \vec{Q} \wedge y \in \sigma(z_1, \vec{z})) \\ & \to \exists z_2.\exists \vec{z}.(z_2 \in P_2 \wedge \vec{z} \in \vec{Q} \wedge y \in \sigma(z_2, \vec{z})) \\ \text{iff} & \vdash \exists z_1.\exists \vec{z}.(z_1 \in P_1 \wedge \vec{z} \in \vec{Q} \wedge y \in \sigma(z_1, \vec{z}) \\ & \qquad \to z_1 \in P_2 \wedge \vec{z} \in \vec{Q} \wedge y \in \sigma(z_1, \vec{z})) \\ \text{iff} & \vdash \exists z_1.\exists \vec{z}.(z_1 \in P_1 \to z_1 \in P_2) \\ \text{if} & \vdash \exists z_1.(z_1 \in P_1 \to z_1 \in P_2) \\ \text{if} & \vdash P_1 \to P_2. \end{array}$$

□

Corollary 27 (Frame Rule)**.** $\vdash \lfloor P \to Q \rfloor \to (\sigma(P) \to \sigma(Q))$

3 Applications

3.1 Axiomatizing transition systems

3.2 Symbolic execution

3.3 Context

Fix a signature (S, Σ). For each sort $s \in S$, introduce an infinite number of *hole* variables, written as $\Box_1, \Box_2, \cdots$. Think of hole variables as normal matching logic vari-

ables, but lie in a disjoint namespace. Extend the grammar by adding the following.

$$\begin{aligned} P ::= & \cdots \\ & \mid \Box \\ & \mid \gamma\Box.P \\ & \mid P[P']. \end{aligned}$$

The sort of $\gamma\Box.P$ is the sort of P. The sort of $P[P']$ is the sort of P, too. Think of $\gamma\Box$ as a binder. Alpha-renaming is always assumed. Patterns of the form $\gamma\Box.P$ are often called *contexts*, denoted by $C, C_0, C_1, \ldots$. The pattern $P[P']$ is often called an *application*. The $\_[\_]$ operator is left associative.

Definition 28. *The context $\gamma\Box.\Box$ is called the identity context, denoted as I. Identity context has the axiom schema $I[P] = P$ where P is any pattern.*

Example 29. $I[I] = I$.

Example 30. *Consider a signature (S, Σ) of a simple imperative programming language, with $S = \{BExp, Pgm\}$, and $\Sigma = \{skip, ite, seq, true, false\}$. Add axiom schemata*

$$ite(C_1[B], P, Q) = (\gamma\Box.ite(C_1[\Box], P, Q))[B]$$

and

$$seq(C_2[P], Q) = (\gamma\Box.seq(C_2[\Box], Q))[P],$$

where P, Q are Pgm patterns, B is a $BExp$ pattern, C_1 is a $BExp$ context, and C_2 is a Pgm context.

Suppose we have the rewrite rules (schemata):

- $C[ite(true, P, Q)] \Rightarrow C[P]$,
- $C[ite(false, P, Q)] \Rightarrow C[Q]$,
- $C[seq(skip, Q)] \Rightarrow C[Q]$.

Example (a). Rewrite $seq(skip, skip)$.

$$\begin{aligned} seq(skip, skip) &= I[seq(skip, skip)] \\ &\Rightarrow I[skip] \\ &= skip. \end{aligned}$$

Example (b). Rewrite $ite(true, P, Q); R$.

$$\begin{aligned} seq(ite(true, P, Q), R) &= seq(I[ite(true, P, Q)], R) \\ &= (\gamma\Box.seq(I[\Box], R))[ite(true, P, Q)] \\ &\Rightarrow (\gamma\Box.seq(I[\Box], R))[P] \\ &= seq(I[P], R) \\ &= seq(P, R). \end{aligned}$$

Definition 31. *Let $\sigma \in \Sigma_{s_1 \ldots s_n, s}$ is an n-arity symbol. We say σ is* active *on its ith argument ($1 \leq i \leq n$), if $\sigma(P_1, \ldots, C[P_i], \ldots, P_n) = (\gamma\Box.\sigma(P_1, \ldots, C[\Box], \ldots, P_n))[P_i]$. Orienting the equation from the left to the right is often called* heating*, while orienting from the right to the left is called* cooling.

Example 32. *Suppose f and g are binary symbols who are active on their first argument. Suppose a, b are constants, and x is a variable. Let $\Box_1$ and $\Box_2$ be two hole variables. Define two contexts $C_1 = \gamma\Box_1.f(\Box_1, a)$ and $C_2 = \gamma\Box_2.g(\Box_2, b)$.*

Because f is active on the first argument,

$$\begin{aligned} C_1[\varphi] &= (\gamma\Box_1.f(\Box_1, a))[\varphi] \\ &= (\gamma\Box_1.f(I[\Box_1], a))[\varphi] \\ &= f(I[\varphi], a) \\ &= f(\varphi, a), \text{ for any pattern } \varphi. \end{aligned}$$

And for the same reason, $C_2[\varphi] = g(\varphi, b)$. Then we have

$$\begin{aligned} C_1[C_2[x]] &= C_1[f(x, a)] \\ &= g(f(x, a), b). \end{aligned}$$

On the other hand,

$$\begin{aligned} g(f(x, a), b) &= g(C_1[x], b) \\ &= (\gamma\Box.g(C_1[\Box], b))[x] \\ &= (\gamma\Box.g(f(\Box, a), b))[x]. \end{aligned}$$

Therefore, the context $\gamma\Box.g(f(\Box, a), b))$ is often called the composition *of C_1 and C_2, denoted as $C_1 \circ C_2$.*

Example 33. *Suppose f is a binary symbol with all its two arguments active. Suppose C_1 and C_2 are two contexts and a, b are constants. Then easily we get*

$$\begin{aligned} f(C_1[a], C_2[b]) &= (\gamma\Box_2.f(C_1[a], C_2[\Box_2]))[b] \\ &= (\gamma\Box_2.((\gamma\Box_1.f(C_1[\Box_1], C_2[\Box_2]))[a]))[b]. \end{aligned}$$

What happens above is similar to curring *a function that takes two arguments. It says that there exists a context C_a, related with C_1, C_2, f and a of course, such that $C_a[b]$ returns $f(C_1[a], C_2[b])$. The context C_a has a binding hole $\Box_2$, and a body that itself is another context C'_a applied to a. In other words, there exists C_a and C'_a such that*

- $f(C_1[a], C_2[b]) = C_a[b]$,
- $C_a = \gamma\Box_2.(C'_a[a])$,
- $C'_a = \gamma\Box_1.f(C_1[\Box_1], C_2[\Box_2])$.

A natural question is whether there is a context C such that $C[a][b] = f(C_1[a], C_2[b])$. In fact, define $C = \gamma\Box_1.\gamma\Box_2.f(C_1[\Box_1], C_2[\Box_2])$. Then

$$\begin{aligned} C[a][b] &= (\gamma\Box_1.\gamma\Box_2.f(C_1[\Box_1], C_2[\Box_2]))[a][b] \\ &= (\gamma\Box_2.f(C_1[a], C_2[\Box_2]))[b] \\ &= f(C_1[a], C_2[b]). \end{aligned}$$