Proof Theory

枫聆

2022 年 7 月 3 日

目录

1 Basic Logic

1.1 Logical Framework

Definition 1 Meta logic is the logic that used to formalize anthor logic

Definition 2 Object logic is the logic that formalized from meta logic.

Definition 3 Implementations of meta logics to represent and reason about object logics are called logical frameworks.

Annotation 4 上面提到的 implementations 实际上就是 languages，例如 Prolog 是这样的一个 language，它以 Horn clause 作为 meta logic. 你要实现的 object logic 可以通过这个 language 来进行 encoding，也就是所谓的 representating，同时还可以在 encoding 基础上进行 reasoning.

1.2 First Order Logic

Example 5 给定一个 statement:

"Every natural number is even or odd, but not both."

如果我们想要构造一个 system 去 accept 这个 statement，我们需要说明 natural number 是什么？even 和 odd 又是什么？如何描述 even 和 odd 是冲突的？这里引出这个 system 需要描述的三个重要部分: objects, their properties, and relations between them. 同时如何描述 every natural number? 这里又涉及到了 quantifiter.

Annotation 6 我遵循描述一个 logical system, 首先定义里面的 terms, 再描述由这些 terms 构成的 formulas.

Definition 7 A signature σ consist of *constant symbols*, *function symbols* and *predicate symbols*.

Definition 8 Let σ-*terms* be a set of terms in first order logic, it defined by the follow inductive process:

1. Each variable $x \in \sigma$-*terms*.
2. Each constant symbol $a \in \sigma$, then $a \in \sigma$-*terms*.
3. If $t_1, \cdots, t_2 \in \sigma$-*terms* and f is k-ary function symbol in σ, then $f(t_1, \cdots, t_k) \in \sigma$-*terms*.

Let $\mathcal{F}_{\sigma\text{-terms}}$ be a set of formulas in first order logic defined by follow inductive process:

1. Given terms $t_1, \cdots, t_k \in \sigma$-*terms* and k-ary predicate symbol P then $P(t_1, \cdots, t_k)$ is a term.
2. All formulas relate to logical connective in propositional logic.
3. If $F \in \mathcal{F}_{\sigma\text{-terms}}$ and x is a variable, then $\forall x.F$ and $\exists x.F$ are both in $\mathcal{F}_{\sigma\text{-terms}}$.

Annotation 9 在理解 propositional logic 的基础上，只要记住几个关键词 *function*, *predicate*, and *quantifier* 就足以刻画 first order logic.

Definition 10 An atomic formula or atom in first-order logic is simply a predicate applied to a tuple of terms; that is, an atomic formula is a formula of the form $P(t_1, \cdots, t_n)$ for P a predicate, and the t_i terms.

Annotation 11 上面 first-order logic 里面 atoms 的定义可以好好想想. 我们可能之前印象中 atoms 只有 constants symbols，现在多了一个 predicates.

Definition 12 An atomic formula is a literal. and if a is an atomic formula, then $\neg A$ is a literal.

Definition 13 A finite set (possibly empty) of literals and logical connectives is called a clause. The empty clause is denoted by $\square$.

Annotation 14 一个 clause 用什么 connective 取决于 context. 注意在 connective 为 disjunction 下的 empty clause 的真值可以总是被解释为 false, 这是因为当你考虑 clause $\square \vee C$ 的时候, 其中 C 是一个任意的 clause, 那么 $\square \vee C$ 的真值实际上就是 C 的真值, 这正好对应了 monoid $(\{true, false\}, \vee)$, 在它里面 $false$ is the neutral element, 所谓 neutral element 就是说对这个 monoid 里面任意一个元素 a 都有 $false \vee a = a$. 当 connective 为 conjunction 的时候, 此时 empty clause 的真值总是可以被解释为 true, 同理.

Definition 15 A literal which contains no variables os called a ground literal.

Definition 16 A clause, each member of which is a ground literal, is called a ground clause. In particular $\square$ is ground clause.

1.3 Logical Formulas Means

Definition 17 An interpretation of a logic formula F is the assignment of meanings to the symbols of F.

Definition 18 If F is true under an interpretation $\mathcal{I}$, then $\mathcal{I}$ is called a model of F

Annotation 19 关于 model 的另一种解释是: a set of ground literals which does not include a complementary pair $(a, \neg a)$.

Definition 20 A formula F is satifiable if it has a model. It is unsatisfiable if it has no model.

Definition 21 A formula F is valid if it is true under every interpretation.

Theorem 22 In classical logic:

$$F \text{ vaild} \leftrightarrow \neg F \text{ unsatisfiable.}$$

1.4 Logical Formula Transformations

Definition 23 A formula F is in prenex normal form if F is composed by a sequence of quantifier followed by a quantifier-free part(called the matrix)

Example 24 例如 $\forall x.\forall y.F(x,y)$，其中 F is quantifier-free，那么这个就是一个 prenex normal form.

Definition 25 A formula F is in negation normal form if all its negation are applied to atoms.

Definition 26 A formula F is in conjunctive normal form (CNF) if it is a conjunction $C_1 \wedge \cdots \wedge C_n$, where $C_1 \wedge \cdots \wedge C_n$, where each C_i is disjunction of literals (i.e., atoms or negated atoms). Each C_i is called a clause.

Definition 27 Let F be a formula in prenex normal form. The skolemization of F consists on replacing every existentially quantifered varible y by a fresh function symbol $f(x_1,\cdots,x_n)$, where $x_1,\cdots,x_n$ are the universally quantified varaibles occurring before y in the prefix of F. Additionally the existential quantification on y is removed. When there are no more existential quantifiers left the formula is skolemized. The f is called Skolem function (or Skolem constant if it is of zero arity) and the term $f(x_1,\cdots,x_n)$ is called Skolem term.

Example 28 例如给定 $\exists x.\forall y.\exists z.P(x,y,z)$，它对应的 skolemization 为 $\forall y.P(c,x,f(y))$.

Theorem 29 (Skolemization does not change sat) The satisfiability of F under skolemization is equivalent.

Proof 考虑任意的 formula $F = \forall x_1.\cdots\forall x_n.\exists y R(x_1,\cdots,x_n,y)$. 这里我们可以拆解一下 model, 定义一个 M, which contains evalutions for every function f in F. 如果 F is satified under M if and only if, for each possible terms for $x_1,\cdots,x_n$ (actually are strings that contains x_i) under the domain of functions in M, there exists a term for y under the domain of functions in M that makes $R(x_1,\cdots,x_n,y)$ is true. 因此根据 axiom of choice, there exists a function f such that $y = f(x_1,\cdots,x_n)$，然后我们把这个 f 再加到 M 上，这样 $\forall x_1.\cdots\forall x_n.R(x_1,\cdots,x_n,f(x_1,\cdots,x_n))$ is also satified under $M \cup f$. □

Annotation 30 Skolemization 里面实际使用了 second-order logical equivalence:

$$\forall x.\exists y.R(x,y) \iff \exists f.\forall x.R(x,f(x))$$

1.5 Satisfiabitity of Logic Formulas

Annotation 31 如果考虑 logic formulas $F = C_1 \wedge \cdots \wedge C_n$，其中 C_i is clause with disjunction, 注意 F 里面的出现的 variables 都是被 universal quantified，这是只是 implicitly 化了. 可以想想一旦 F 里面存在某个 empty clause，那么显然 F is unsat, 因为 empty clause is always false.

Annotation 32 现在只记录一下当下我自己的思考，未来会系统补充.

- 关于 connectives 对 sat 的影响:
  - 若 $F = F_1 \vee F_2$ and F is unsat, 那么需要 F_1 is unsat 和 F_2 is unsat.
  - 若 $F = F_1 \wedge F_2$ and F is unsat，那么需要 F_1 is unsat 或者 F_2 is unsat.

  可以看到它和 connectives 的关系好像是反过来了.

1.6 Satisfiability of Sets of Formulas

Definition 33 If v is a valuation, this is, a mapping from the atoms to the set $\{t, f\}$.

Definition 34 [4] Let Σ denote a set of well-formed formulas and t a valuation. Define

$$\Sigma^t = \begin{cases} T & \text{if for each } \beta \in \Sigma, \beta^t = T \\ F & \text{otherwise} \end{cases}$$

When $\Sigma^t = T$, we say that t satisfies Σ. A set Σ is satisfiable iff there is some valuation t such that $\Sigma^t = T$.

Definition 35 Let Σ be a set of formulas, and let α be a formula, we say that

1. α is a logical consequence of Σ, or
2. Σ (semantically) entails α, or
3. $\Sigma \models \alpha$,

if and only if for all truth valuations t, if $\Sigma^t = T$ then also $\alpha^t = T$. We write $\Sigma \not\models \alpha$ for there exists a truth valuation t such that $\Sigma^t = T$ and $\alpha^t = F$.

Annotation 36 For example, $\Sigma = \{p_1, p_2, \cdots, p_n\}$ could be a set of premises and let α could be the conclusion that we want to derive.

1.7 Classic Propositional Modal Logic

Definition 37 [8]Let Σ be a set of propositional letters or atomic propositions. The set $F_P(\Sigma)$ of formulas of classical propositional modal logic is the smallest set with:

1. If $A \in \Sigma$ is a propositional letter, then $A \in F_P(\Sigma)$;
2. If $\phi, \psi \in F_P(\Sigma)$, then $\neg\phi, (\phi \wedge \psi), (\phi \vee \psi), (\phi \to \psi) \in F_P(\Sigma)$;
3. If $\phi \in F_P(\Sigma)$, then $(\Box\phi), (\Diamond\phi) \in F_P(\Sigma)$.

Definition 38 Let $\mathcal{S}$ be a system of modal logic, this is $F_P(\Sigma)$ with a set of axioms and rules. If axioms and rules as follow

$$\text{all propostional tautologies} \qquad \text{(P)}$$

$$\Box(\phi \to \psi) \to (\Box\phi \to \Box\psi) \qquad \text{(Kripke axiom)}$$

$$\Box\phi \to \phi \qquad \text{(T)}$$

$$\Box\phi \to \Box\Box\phi \qquad \text{(4)}$$

$$\frac{\phi \quad \phi \to \psi}{\psi} \qquad \text{(modus ponens)}$$

$$\frac{\phi}{\Box\phi} \qquad \text{(Gödel)}$$

We call it modal logic $\mathcal{S}4$.

Annotation 39 Kriple axiom 原本的形式应为

$$\Box\phi \wedge \Box(\phi \to \psi) \to \Box\psi$$

上面是它经常用的等价形式. Axiom T 是指若 if we ϕ is necessary, the we have ϕ. Axiom 4 是指 ϕ is necessary, 那么命题"ϕ is necessary" is necessary, 有点别扭, 举个形象的例子如果 box 是指某个人知道某件事, 假设我知道 A *true*, 那么我肯定知道我知道 A *true*. 最后一个叫 Gödel translation, 它将 intuitionistic logic 里面的 formulas 转换到 modal logic 里面.

Definition 40 Let $\mathcal{S}$ be a system of modal logic. For a formula ψ and a set of formulas Φ, we write $\Phi \vdash_{\mathcal{S}} \psi$ and say that ψ can be derived from Φ(or is provable from Φ), iff there is a proof of ψ that uses only the formulas of Φ and the axioms and proof rules of $\mathcal{S}$. That is, we define $\Phi \vdash_{\mathcal{S}} \psi$ inductively as:

$$\Phi \vdash_{\mathcal{S}} \psi$$

iff $\psi \in \Phi$ or there is an instance

$$\frac{\phi_1 \quad \cdots \quad \phi_n}{\psi}$$

of a proof rule of $\mathcal{S}$ with conclusion ψ and some number $n \geq 0$ of premises such that for all $i = 1, 2, \cdots, n$, the premises ϕ_i is derivable, i.e:

$$\Phi \vdash_{\mathcal{S}} \phi_i$$

When the case $n = 0$ corresponds to axioms.

Annotation 41 现在以 $\Box$ 表示 provable 的视角来看待前面提到的 axioms. 首先是

$$\Box(\phi \to \psi) \to (\Box\phi \to \Box\psi) \quad \text{(Kripke axiom)}$$

若 $\phi \to \psi$ is provable 且 ϕ is provable, 那么则 ψ is provable.

$$\Box\phi \to \phi \quad \text{(T)}$$

若 ϕ is provable, 那么 ϕ should be true.

$$\Box\phi \to \Box\Box\phi \quad (4)$$

若 ϕ is provable, 那么 ϕ should be provably provable, 也就是我们肯定知道存在一个 proof.

$$\frac{\phi}{\Box\phi} \quad \text{(Gödel)}$$

若 ϕ is proven, 那么 ϕ should be provable.

Definition 42 A Kripke frame (W, ρ) consists of a non-empty set W and a relation $\rho \subseteq W \times W$ on worlds. The element of W are called possible worlds and ρ is called accessibility relation.

Definition 43 A Kripke structure $K = (W, \rho, v)$ consists of Kripke frame (W, ρ) and a mapping $v : W \to \Sigma \to \{true, false\}$ that assigns truth-values to all the propositional letters in all worlds.

Definition 44 Given a Kriple structure $K = (W, \rho, v)$, the interpretation $\vDash$ of modal formulas in worlds s is defined as

- $K, s \vDash A$ iff $v(s)(A) = true$;
- $K, s \vDash \phi \wedge \psi$ iff $K, s \vDash \phi$ and $K, s \vDash \psi$;
- $K, s \vDash \phi \vee \psi$ iff $K, s \vDash \phi$ or $K, s \vDash \psi$;
- $K, s \vDash \neg\phi$ iff it is not the case that $K, s \vDash \phi$;

- $K, s \vDash \Box\phi$ iff $K, t \vDash \phi$ for all worlds t with $s\rho t$;
- $K, s \vDash \Diamond\phi$ iff $K, t \vDash \phi$ for some worlds t with $s\rho t$.

Annotation 45 最后两个关于 modality $\Box$ 和 $\Diamond$ 定义是最重要的，它们借助 accessible possible world 来 make sense. 可以通过它们的 nesting 形式来描述更长的路径即 $\Box\Box, \Diamond\Diamond, \Box\Diamond$.

Definition 46 Given a Kripke structure $K = (W, \rho, v)$, formula ϕ is vaild in K, written $K \vDash \phi$, iff $K, s \vDash \phi$ for all worlds $s \in W$.

Definition 47 (local consequence) Let ψ be a formula and Φ a set of formulas. Then we write $\Phi \vDash_l \psi$ if and only if, for each Kripke structure $K = (W, \rho, v)$ and each world $s \in W$, we have $K, s \vDash \Phi$ implies $K, s \vDash \psi$.

Definition 48 (global consequence) Let ψ be a formula and Φ a set of formulas. Then we write $\Phi \vDash_g \psi$ if and only if, for each Kripke structure $K = (W, \rho, v)$, if for all world $s \in W : K, s \vDash \Phi$, then for all world $s \in W : K, s \vDash \psi$.

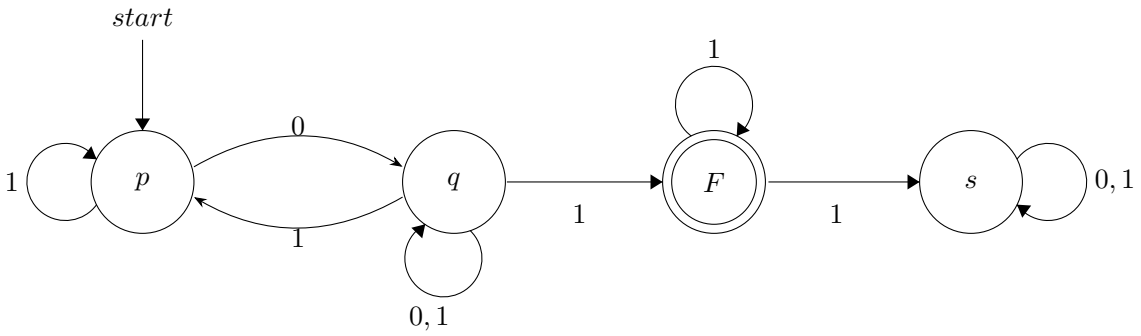
Annotation 49 local consequence 和 global consequence 的区别就是 assumption 是在某个 world 里面还是在所有的 worlds 里面.

Definition 50 A formula ϕ is vaild or a tautology, iff $\emptyset \vDash_l \phi$, which we write $\vDash \phi$. A set of formulas Φ is called satisifable, iff there is a Kripke structure K and a world s with $K, s \vDash \Phi$.

Lemma 51 (local deduction theorem) For formulas ϕ, ψ we have

$$\phi \vDash_l \psi \iff \vDash_l \phi \to \psi.$$

Annotation 52 (view of finite automata) 对于 Kripke frame 的第一反应应该是 finite automata, 但是对于一个给定的 finite automata 我们还需要一些额外的说明. 例如



每一个 state 里面存在一个 proposition，它表示这个 proposition is hold at this state，自然地 states 就变成了 possible worlds. state 现在可以接受多个输入 $\{0,1\}$，那么这里就表示我们有两个 relations ρ_0 和 ρ_1，对应我们需要两个 pair 来构建不同的 modality $(\Box_0, \Diamond_0)$ 和 $(\Box_1, \Diamond_1)$，它们都是用于描述某个 state 的 successor. 因此这里可以对应上一个 Kripke structure, 对上图我们可以列举几个 vaild formula.

$$\begin{aligned} &K \vDash \neg\Diamond_0 F && \text{does not end with 0} \\ &K \vDash p \to \Diamond_0 p && p \text{ has a 1-loop} \\ &K \vDash \Diamond_0\ true && \text{never stuck with input 0} \\ &K \vDash \Diamond_1\ true && \text{never stuck with input 1} \end{aligned}$$

再看一个稍微复杂一点

$$K \vDash F \to \Diamond_0(\neg\Diamond_0 F \wedge \neg\Diamond_1 F)$$

它意思如果某个状态 σ 下 F is hold, 那么 σ accept 0 的 successors $\{s_i\}$ 中每个 s_i 的 successors 都无法 hold F，显然这是成立的.

Definition 53 A system $\mathcal{S}$ of proof rules and axioms of modal logic is sound iff, for all formulas ψ and all sets of formulas Φ:

$$\Phi \vdash_S \psi \text{ implies } \Phi \vDash_g \psi$$

Annotation 54 上述 soundness 实际在建立关于 axiomatic modal logic 和 semantic modal logic 之间的一座桥，这座桥需要每一个 axiom make sense.

Lemma 55 Kripke axiom $\Box(\phi \to \psi) \to (\Box\phi \to \Box\psi)$ is sound.

Proof 首先给定任意一个 Kripke structure K. 我们需要证明

$$K, s \vDash \Box(\phi \to \psi) \to (\Box\phi \to \Box\psi).$$

因此假设其前提

$$\begin{aligned} &K, s \vDash \Box(\phi \to \psi) \\ &K, s \vDash \Box\phi \end{aligned}$$

那么对应所有满足 $s\rho t$ 的 successor t, 都有

$$\begin{aligned} &K, t \vDash \phi \to \psi \\ &K, t \vDash \phi \end{aligned}$$

自然地这里有 $K, t \vDash \psi$，于是 $K, s \vDash \Diamond\psi$. □

Lemma 56 Gödel Rule $\dfrac{\phi}{\Box\phi}$ is sound.

Proof 注意这里的结论是建立在 global assumption 上的，即 $K, s \vDash \phi$ for any $s \in W$，证明过程是显然的. □

Lemma 57 A Kripke frame (W, ρ) is reflexive, that ρ is reflexive, if and only if $K, s \vDash \Box q \to q$ for all Kripke structures $K = (W, \rho, v)$.

Proof ($\Rightarrow$) 若 (W, ρ) is reflexive, 这是显然的.

($\Leftarrow$) 若 $K, s \vDash \Box q \to q$ for all Kripke structures $K = (W, \rho, v)$. 假设存在一个 r such that $(r, r) \notin \rho$, 构造一个比较巧妙地 valuation v

$$v(s)(q) = \begin{cases} true & \text{if } r \rho s \\ false & \text{otherwise} \end{cases}$$

那么显然有 $K, r \vDash \Box q$，根据前提这里有 $K, r \vDash q$，而根据 valuation 这里就 r 存在一个 successor 是它自己，即 (r, r) 与假设矛盾. □

Lemma 58 A Kriple frame (W, ρ) is transitive, that ρ is transitive, if and only if $K, s \vDash \Box q \to \Box\Box q$ for all Kripke structures $K = (W, \rho, v)$.

Proof ($\Rightarrow$) 若 (W, ρ) is transitive, 给定 $K, s \vDash \Box q$，对于 s 的任意一个 successor $t (s \rho t)$ 则有 $K, t \vDash p$，进一步对 t 的任意一个 successor $r (t \rho r)$，考虑 transitive $s \rho r$，那么有 $K, r \vDash p$. 由于 t 和 r 的任意性，因此 $K, s \vDash \Box\Box p$.

($\Leftarrow$) 若 Kriple frame 满足对任意的 valuation v 都有 $K, s \vDash \Box q \to \Box\Box q$. 假设 (W, ρ) 不是 transitive，那么存在 $r_1, r_2, r_3 \in W$ such that $r_1 \rho r_2, r_2 \rho r_3$ and $(r_1, r_3) \notin \rho$. 构造一个 valuation v

$$v(s)(q) = \begin{cases} true & \text{if } r_0 \rho s \\ false & \text{otherwise} \end{cases}$$

那么 $K, r_0 \vDash \Box q$，但是因为 $(r_0, r_3) \notin \rho$，因此 $K, r_0 \nvDash \Box\Box q$，和假设前提矛盾了. □

Annotation 59 这座需要两边的支撑一样高，给定特定 axiomatic modal logic, 我们得到找到与之对应的 semantic modal logic，我们的手法就是 sketch it from basic Kripke frame. 当我们尝试构造了一部分之后，我们需要让其 make sense，上述 lemma 利用 formula 来 characterize 是一个不错的选择.

Definition 60 (characterization)Let C be a class of Kripke frames and ϕ a formula in modal logic. Formula ϕ characterizes C, if for every Kripke frame (W, ρ):

$$(W, \rho) \in C \text{ iff for each } v : K, s \vDash \phi \text{ holds for } K = (W, \rho, v).$$

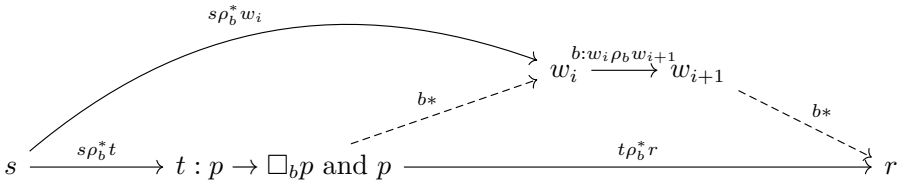
Theorem 61 (soundness of $\mathcal{S}4$)The Kriple proof rules for $\mathcal{S}4$ are sound for the class of reflexive and transitive frames.

Theorem 62 The conjunction of the following two multimodal formulas

$$\Box_a p \to (p \wedge \Box_a \Box_b p)$$
$$\Box_a(p \to \Box_b p) \to (p \to \Box_a p)$$

characterizest the class of all multimodal kripke frames (W, ρ_a, ρ_b) such that ρ_a is the reflxive, transitive closure of ρ_b.

Proof ($\Leftarrow$) 如果 $(W, \rho_a, \rho_b)$is Kripke frame where ρ_a is the reflexive, transitive closure of ρ_b. 对于一个 formula 只要注意到 $\Box_a \Box_a p \to \Box_a \Box_b p$ 即可，可以从需要考虑的 successors 数量来证明. 对于第二个 formula, 先给一个思考图



这里 $\rho_a = \rho_b^*$. 这里证明手法是

$$\Box_a(p \to \Box_b p) \to \Box_a(p \to \Box_a p) \text{ and } \Box_a(p \to \Box_a p) \to (p \to \Box_a p)$$

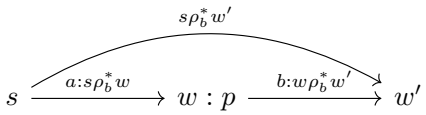
最重要是证明第一个 implication，第二个 implication 是前面已经证明过的 reflexive. 对于第一个 implication 它描述的是首先给出前提 (1)$\Box_a(p \to \Box_b p)$ 即 $s\rho_b^* t$. 然后我们想要将 t 中 $p \to \Box p$ 扩展至 $p \to \Box_b^* p$，因此再给一个假设前提 t holds p，我们来考察 $\Box_b^* p$ 是否成立即 $t\rho_b^* r$. 这里我们需要分解 $t\rho_b^* r$ 使其为 $w_i \rho_b w_{i+1}$ for all $i < n$，其中 $w_0 = t$ 和 $w_n = r$. 利用数学归纳法证明 $K, w_i \vdash p$，这里就不详细描述了，和后面一个证明过程类似，但是说明几点: (1)$s\rho_b^* w_i$ 送来了 $p \to \Box_b p$ (2) 假设前提保证了 $K, w_i \vDash p$. 因此 $K, w_{i+1} \vDash p$.

($\Rightarrow$) 如果 $(W, \rho_a, \rho_b)$is Kripke frame such that above formulas are vaild in it for any valution v. 我们得证明 $\rho_a = \rho_b^*$. 这种证明两个集合相等的手法，还是用两边证.

先证$\rho_a \subseteq \rho_b^*$. 取任意的 $(s,t) \in \rho_a$，我们得证明 $(s,t) \in \rho_b^*$. 还是构造一个特殊的 valuation

$$v(w)(q) = \begin{cases} true & \text{if } (s,w) \in \rho_b^* \\ false & \text{otherwise} \end{cases}$$

我们的思路是首先证明第二个 formula 的前提 (1) $\Box_a(p \to \Box_b p)$，从而得到对应的 conclusion (2) $(p \to \Box_a p)$，由给定的 v 结合 ρ_b^* 的 reflexive 性质，自然地有 $K, s \vDash p$，在使用一下 (2) 得到 $K, t \vDash p$, 这样就有 $(s,t) \in \rho_b^*$. 证明 (1) 思路是依然是假设前提: 给定 $s\rho_a w$ 且 $K, w \vDash p$，实际上 $(s,w) \in \rho_b^*$. 考虑下面的思考过程

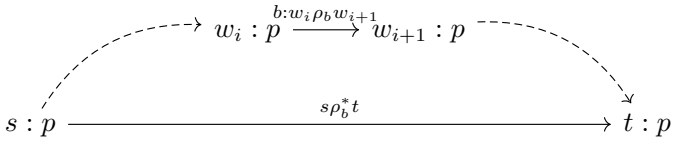


另外又给了一个 w' 满足 $w\rho_b w'$，再根据 ρ_b^* 的 transitive 得到 $s\rho_b^* w'$，从而 $K, w' \vDash p$.

再证$\rho_a \supseteq \rho_b^*$. 取任意的 $(s,t) \in \rho_b^*$，我们要证明 $(s,t) \in \rho_a$. 依然构造一个类似的 valuation

$$v(w)(q) = \begin{cases} true & \text{if } (s,w) \in \rho_a \\ false & \text{otherwise} \end{cases}$$

我们的思路: 由于我们构造地特别的 v 有 $K, s \vDash \Box_a p$, 借助命题中的第一个 formula 得到对应的 conclusion (1) $K, s \vDash p \wedge \Box_a \Box_b p$. 考虑下面的思考过程



我们考虑将 $s\rho_b^* t$ 拆开，设 $w_i \rho_b w_{i+1}$ for all $i < n$，其中 $w_0 = s$ 和 $w_n = t$，这是可以做到的，考虑 closure 的构造过程. 再用一下数学归纳法证明 $K, w_i \vDash p$，在 $i = 0$ 显然是成立的，假设 w_i 成立，那么根据 v 即有 $s\rho_a w_i$，再利用一下 (1) 可以得到 $K, w_i \vDash \Box_b p$，因此 $K, w_{i+1} \vDash p$. 最终 $K, t \vDash p$，那么 $(s,t) \in \rho_a$. □

Annotation 63 回顾上面的证明手法，我们如果想要刻画两个 possible worlds 是否存在某种关系，例如 $(r,t) \overset{?}{\in} \rho$，我们可以额外借助一个 formula p 和 valuation v，仅使得所有 w 满足 $r\rho w$ 都 hold p. 这样如果我们能利用额外和 p 相关的条件间接证明 t holds p, 那么就可以证明 $s \to t$. 我们应该意识到 relations 是 Kripke frame 固有的性质，与 valuation 无关因此这里我们可以任意的定义它.

2 Natural Deduction

Remark 64 Natural deduction is a kind of proof calculus in which logical reasoning is expressed by inference rules closely related to the "natural" way of reasoning.

2.1 Judgments and Propositions

Definition 65 A *judgment* is somthing we may know, this is, an object of knowledge. A judgment is *evident* if we in fact know it.

Annotation 66 "A is false" (see classical logic), "A is true at time t" (see temporal logic), "A is necessarily true" or "A is possibly true" (see modal logic), "the program M has type " (see programming languages and type theory), "A is achievable from the available resources" (see linear logic).

2.2 Introduction and Elimination

Definition 67 Inference rules that introduce a logical connective is the conclusion are known as *introduction rules*. i.e., to conclude "A and B *true*" for propositions A and B, one requires evidence for "A *true*" and B *true*. As an inference rule:

$$\frac{A\ true \quad B\ true}{A \wedge B\ true}\ \wedge I$$

Here $\wedge I$ stands for "conjunction introduction".

Annotation 68 实际上面的 inference rule 的 general form 应该是

$$\frac{A\ prog \quad B\ prog \quad A\ true \quad B\ true}{A \wedge B\ true}\ \wedge I$$

这里才能帮助后面的 $\vDash$ make sense.

Definition 69 Inference rules that describe how to deconstruct information about a compound proposition into information about its consitiuents are elimination rules. i.e., from $A \wedge B$ *true*, we can conclude A *true* and B *true*:

$$\frac{A \wedge B\ true}{A\ true}\ \wedge E_L \qquad \frac{A \wedge B\ true}{B\ true}\ \wedge E_R$$

Annotation 70 The meaning of conjunction is determinded by its *verifications*.

2.3 Hypothetical Derivations

Definition 71 A *hypothetical judgment* is $J_1, \cdots, J_n \vdash J$, where judgments $J_1, \cdots, J_n$ are unproved assumptions, and the judgment J is the conclusion. A *hypothetical deduction*(derivation) for $J_1, \cdots, J_n \vdash J$ has the form

$$\begin{array}{c} J_1 \quad \cdots \quad J_n \\ \vdots \\ J \end{array}$$

which means J is derivable from $J_1, \cdots, J_n$.

Annotation 72 上面的 $J_1, \cdots, J_2$ 都可以替换成关于 J_i 的一个 hypothetical derivation.

Definition 73 In the natural deduction calculus, an assumption is discharged when the conclusion of an inference does not depend on it, although one of the premises of the inference does[1].

Annotation 74 Once the appropriate rules have been completed, these are known as discharged assumptions, and are not included in the pool of assumptions on which the conclusion of the rule depends[3].

Annotation 75 hypothetical derivation 要求最后的 conclusion 依赖的 poof of assumptions 不是空的.

Theorem 76 Deduction theorem

$$T, P \vdash Q \iff T \vdash P \to Q$$

.

Annotation 77 在 deduction theorem 中我们注意到第一个 hypothetical judgment 里面的 antecedent Q 被去掉了，在第二个 hypothetical judgment 的 succedent 里面作为一个 implication 的 antecedent 出现了，这里我们就可以说 assumption Q is discharged，即现在的 conclusion 已经不依赖它了. 那么我们是如何构造 deduction theorem 里面的 implication 的呢？下面接着看

Definition 78 (implication) If B is true under the assumption that A is true, formly written $A \supset B$. The corresponed introduction and elimination rule as follow

$$\frac{\begin{array}{c}\overline{A\ true}^{\,u} \\ \vdots \\ B\ true\end{array}}{A \supset B\ true}\ \supset I^u \qquad \frac{A \supset B\ true \quad A\ true}{B\ true}\ \supset E$$

Annotation 79 Why indexed u In the introduction rule, the antecedent named u is discharged in the conclusion. This is a mechanism for delimiting the scope of the hypothesis: its sole reason for existence is to establish "$B\ true$"; it cannot be used for any other purpose, and in particular, it cannot be used below the introduction.

上面这段话出自 natural deduction 的 wiki，这个 $u$scope 了 assumption $A\ true$ 的开端，因为 $A \supset B$ 并不依赖 $A\ true$, 它描述只是 if $A\ true$ then $B\ true$. 同时最后的 introduction rule 会将这个 assumption $A\ true$ discharged 掉，表示 scope 在这里已经结束了. 而 implication rule 会将上述 derivation 直接总结得到一个结论，即

$$A \vdash B \Rightarrow \cdot \vdash A \to B.$$

Example 80 Considering the following proof of $A \supset (B \supset (A \wedge B))$

$$\dfrac{\dfrac{\dfrac{\dfrac{\overline{A\ true}^{\,u} \quad \overline{B\ true}^{\,w}}{A \wedge B\ true}\wedge I}{B \supset (A \wedge B)\ true} I^{w}}{A \supset (B \supset (A \wedge B))\ true} I^{u}.$$

这整个 derivation 不是 hypothetical 的，因为两个 assumptions $A\ true$ 和 $B\ true$ 都已经被 discharged，因此它实际上一个 complete proof!

Definition 81 (disjunction) The elimination rule for disjunction:

$$\dfrac{A \vee B\ true \quad \begin{array}{c}\overline{A\ true}^{\,u}\\ \vdots \\ C\ true\end{array} \quad \begin{array}{c}\overline{B\ true}^{\,w}\\ \vdots \\ C\ true\end{array}}{C\ true} \vee E^{u,w}$$

both assumption u, w are discharged at the disjunction elimination rule.

Definition 82 The falsehood elimination rule:

$$\dfrac{\bot\ true}{C\ true}\ \bot E$$

Annotation 83 falsehood elimination 的意义在哪？首先你应该主要到一个特殊等价命题 $A \vee \bot = A$，从 $\vee$ 的 introduction rule 来看这意味 $\bot\ true \vdash A\ true$，由于 A 是任意的，因此我们得到了 $\bot\ true \vdash C\ true$.

2.4 Harmony

Definition 84 Local soundness shows that the elimination rules are not strong: no matter how we apply eliminations rules to the result of an introduction we cannot gain any new information.

Definition 85 Local completeness shows that the elimination rules are not weak: there is always a way to apply elimination rules so that we can reconstitute a proof of the original proposition from the the results by apply intruduction ruls.

Annotation 86 local soundness 告诉你通过 elimination 得到的东西不会比你已经知道的东西强 (not strong), 而 local completeness 告诉你可以利用通过 elimination 得到的东西来构造你原本你已经知道的东西 (not weak).

Definition 87 Given two deduction of same judgment, we use the notion

$$\begin{matrix}\mathcal{D}\\ A\ true\end{matrix} \Longrightarrow_R \begin{matrix}\mathcal{D}'\\ A\ true\end{matrix}$$

for the local reduction of a deduction $\mathcal{D}$ to another deduction D' of same judgement $A\ true$. Simliarly, we have local expansion

$$\begin{matrix}\mathcal{D}'\\ A\ true\end{matrix} \Longrightarrow_E \begin{matrix}\mathcal{D}\\ A\ true\end{matrix}$$

Definition 88 (substitution Principle) If

$$\begin{matrix}\overline{A\ true}\ u\\ \mathcal{E}\\ C\ true\end{matrix}$$

is a hypothetical proof of $\mathcal{C}\ true$ under the undischarged hypothesis $A\ true$ labelled u, and

$$\begin{matrix}\mathcal{D}\\ A\ true\end{matrix}$$

is a proof of $A\ true$ then

$$\begin{matrix}\dfrac{\mathcal{D}}{A\ true}\ u\\ \mathcal{E}\\ C\ true\end{matrix}$$

is our notation for substituting $\mathcal{D}$ for all uses of the hypothesis labelled u in $\mathcal{E}$. This deduction, also sometime written as $[\mathcal{D}/u]\mathcal{E}$ no longer depends on u.

Example 89 If given a elimination rule of disjunction as follow

$$\frac{A \vee B\ true}{A\ true}\ \vee E_L$$

The rule a little bit stronger, since we would not be able to reduce

$$\dfrac{\dfrac{B\ true}{A \vee B\ true}\ \vee I_R}{A\ true}\ \vee E_L$$

As u can see it's not local soundness.

2.5 Verifications and Uses

Definition 90 a verification should be a proof that only analyzes the consitituents of a proposition.

Annotation 91 [9] 在 natural deduction 中由于 local reduction 的存在，可能会让一个证明过程变得非常的冗余，例如在证明 conjunction commutativity

$$\dfrac{\dfrac{\dfrac{\dfrac{\overline{A \wedge B\ true}^{\,u}}{A\ true}\wedge E_1 \quad \dfrac{\overline{A \wedge B\ true}^{\,u}}{B\ true}\wedge E_2}{A \wedge B\ true}\wedge I}{B\ true}\wedge E_2 \quad \dfrac{\dfrac{\overline{A \wedge B\ true}^{\,u}}{A\ true}\wedge E_1}{B \wedge A\ true}\wedge I}{(A \wedge B) \supset (B \wedge A)\ true}\supset I^u$$

其中左上角的 local reduction 显然是冗余的. 这样对于谈论某个具体 proposition 的 proof 时就会出现问题，因为 the shape of proof is not decidable. 同时我们也希望未来能够设计出一个 tool 用于 deviating proofs automatically，也就是 search proof automatically. 因此从 natural deduction 上诞生了一个新的 calculus，它会在 syntax level 上来施加一些限制，借此限制 the shape of proof. 最后我们将证明这个 calculus 引入的 restrictions 不会产生 side-effect.

Definition 92 Writing $A\uparrow$ for the judgment "A has a verification". Naturally, this should mean that A is true, and that the evidence for that has a special form.

Definition 93 Writing $A\downarrow$ for the judgment "A may be used". $A\downarrow$ should be the case when either $A\ true$ is a hypothesis, or A is deduced from a hypothesis via elimination rules.

Annotation 94 我觉得下述两种理解方式更为明确易懂

- $A\uparrow$ denotes that we are searching for a verfication of A;
- $A\downarrow$ denotes that we are allowed to use A.

Annotation 95 上述两个 definitions 里面隐藏着非常重要但有点不正式的结论:If A has a vertification then A true, 反之依然. 后面我们将形式化地证明它们.

Definition 96 For conjunction.

$$\dfrac{A\uparrow \quad B\uparrow}{A \wedge B\uparrow}\wedge I \qquad \dfrac{A \wedge B\downarrow}{A\downarrow}\wedge E_L \qquad \dfrac{A \wedge B\downarrow}{B\downarrow}\wedge E_R$$

Definition 97 For implication

$$\dfrac{\begin{array}{c}\overline{A\downarrow}^{\,u}\\ \vdots\\ B\uparrow\end{array}}{A \supset B\uparrow}\supset^u \qquad \dfrac{A \supset B\downarrow \quad A\uparrow}{B\downarrow}\supset E$$

Annotation 98 (why implication) In order to have a verification of $A \supset B$, we need a proof of B and we are given an assumption A to work with. Therefore, we will need a verification of B and we are allowed to use A.

When using an implication statement in a proof, we need to show that the antecedent holds, so we need a verification of it. Only then we are allowed to use the consequent.

Example 99

$$\dfrac{(A \supset A) \supset B \; ? \quad \dfrac{\dfrac{A \; ?}{A \; ?}}{A \supset A \; ?}}{\dfrac{B \; ?}{((A \supset A) \supset B) \supset B \uparrow}}$$

Example 100

$$\dfrac{\dfrac{\dfrac{\overline{A \wedge B \; true}^{\,u}}{A \; true} \wedge E_L}{(A \wedge B) \supset A \; true} \supset I^u$$

那么它对应上 verification 和 use

$$\dfrac{\dfrac{A \wedge B \downarrow}{A \downarrow} \wedge E_L}{(A \wedge B) \supset A \; ?} \supset I^u$$

一切都非常奇怪，这个 verification 和 use 到底是怎样对应 truth? 从前面两个例子都可以清晰地感觉到一个阻力，即

$$\begin{array}{c} A \downarrow \\ ??? \\ A \uparrow \end{array}$$

就是当我们在 use A 的时候，实际上存在一个 A has a verification.

Definition 101 For disjunction

$$\dfrac{A \uparrow}{A \vee B \uparrow} \vee I_L \qquad \dfrac{B \uparrow}{A \vee B \uparrow} \vee I_R \qquad \dfrac{A \vee B \downarrow \quad \begin{array}{c}\overline{A \uparrow}^{\,u} \\ \vdots \\ C \uparrow\end{array} \quad \begin{array}{c}\overline{B \downarrow}^{\,w} \\ \vdots \\ C \uparrow\end{array}}{C \uparrow} \vee E^{u,w}$$

Definition 102 For truth and falsehood.

$$\dfrac{}{\top \uparrow} \top I \qquad \dfrac{\bot \downarrow}{C \uparrow} \bot E$$

Annotation 103 $\bot \downarrow$ signifies a contradiction from our hypotheses.

Annotation 104 the elimination rule of disjunction and falsehood 里面出现 conclusion $C\uparrow$ 也很奇怪，为什么不是 $C\downarrow$?

Definition 105 For atomic propositions.

$$\frac{P\downarrow}{P\uparrow}\downarrow\uparrow.$$

Annotation 106 当引入上述的 arrow switch 之后我们可以回答前面的种种问题了. 首先是 example 99，假设其中的 A, B 都是 atomic proposition，则

$$\dfrac{\dfrac{\dfrac{}{(A\supset A)\supset B\downarrow}w \quad \dfrac{\dfrac{\dfrac{}{A\downarrow}u}{A\uparrow}\downarrow\uparrow}{A\supset A\uparrow}\supset I^u}{B\downarrow}\downarrow\uparrow}{((A\supset A)\supset B)\supset B\uparrow}\supset I^w$$

同时如果将 implication emilination 的 premise 换成 $A\downarrow$，在找 $A\supset A\downarrow$ 的 proof 时就被卡住了. example 100类似. 那么有一个很自然的问题这个 arrow switch 能不能推广到任意的 propositions 上呢? 本质上是没有问题的，例如

$$\dfrac{\dfrac{A\supset A\downarrow \quad \dfrac{\dfrac{}{A\downarrow}u}{A\uparrow}\downarrow\uparrow}{A\downarrow}\downarrow\uparrow}{A\supset A\uparrow}\supset I^u$$

但是这样的语法又会使得 proof search space 变大，并不符合我们的初衷，因此我们只将 arrow switch 放在的了 atomic proposition 上，这样做的后果你也可以看到，需要将 connectives 都展开.

再来思考另外一种 arrow switch

$$\frac{F\downarrow}{F\uparrow}\uparrow\downarrow$$

这个人在本质上也是没有问题的，当我们有一个关于 F 的 verification，我们当然可以 use it. 但是引入它同样会造成我们的 proof search space，就像 classical logic 中的 tautologies，我们可以在任何 proof 中使用它，但是有时候是没有意义的. 同在 emilinotation rule of disjunction and falsehood 中的 conclusion 中我们都是使用的 verification，而不是 use，也是为了防止后续使得我们的 proof 变得复杂.

Theorem 107 (Global Soundness) If $A\uparrow$ and $\begin{array}{c}A\downarrow\\ \vdots\\ C\uparrow\end{array}$ then $C\uparrow$

Annotation 108 Global Soundness 意味着如果 if the verification formula of C under the verification formula A，那么在 C 中使用 A，并不会得到任何其他 new informations.

Theorem 109 (Global Completeness) If $A\downarrow$, then $A\uparrow$.

Annotation 110 Global completeness 意味着如果我们确定 formula A 在某种情况下可以使用，那么在相同的 assumptions 下我们可以推导出一个关于它的 verification. 这在前面关于 $\downarrow\uparrow$ 转换规则的 annotation 中已经见识过一个特殊例子了.

2.6 Notational Definition

Definition 111 A notational definition gives the meaning of the general form of a proposition in terms of another proposition whose meaning has already been defined.

Example 112 We can define logical equivalence, written $A \equiv B$ as

$$(A \supset B) \wedge (B \supset A).$$

Example 113 We can define negation $\neg A$ as

$$\neg A = (A \supset \bot) \Longrightarrow \dfrac{\begin{array}{c} A \\ \vdots \\ \bot \end{array}}{\neg A}\ \neg I$$

We also can give the introduction rule of falsehood.

$$\dfrac{\neg A \quad A}{\bot}\ \bot\, I$$

so $\bot$ actually means any contradictions. moreover double negation is coming.

Annotation 114 notaional definition 可以看做用已有的东西构造出一些东西. 与之对应的是我们可以直接符号化的给出某个新的定义，称之为 symbolic definition.

2.7 Soundness and Completeness of Native Natural Deduction

Definition 115 [5]Soundness of natural deduction means that the conclusion of proof is always a logical consequence of the premises. That is

$$\text{If } \Sigma \vdash \alpha, \text{then } \Sigma \vDash \alpha.$$

Definition 116 Completeness of natural deduction means that all logical consequences in propositional logic are provable in natural deduction. That is,

$$\text{If } \Sigma \vDash \alpha, \text{then } \Sigma \vdash \alpha.$$

Annotation 117 其中 $\Sigma \vdash \alpha$，表示存在一个以 Σ 作为 premise 得到 conclusion 为 α 的 proof. 而 $\Sigma \vDash \alpha$，就考虑两端的 proposition 加上 truth-falsehood 了，即如 $\Sigma^t = True$ 则有 $\alpha^t = True$.

对于 soundness 的证明，我们需要根据 α 的结构来做归纳，而后再考虑赋予其 true/false 来考虑. 这里记录一下对于结构归纳它是怎样对应一般归纳法命题 $P(n)$ 结构上，这里的 n 应该对应 α 的 bottom-up derivation 里面的 maximum depth of line.

而对于 completeness 的证明，相对来说会复杂一点. 我们需要下面 3 个 lemma. 有一个疑问不引入 negation 是不是还说明不了 completeness?

Lemma 118 If $\Sigma = \{\alpha_0, \alpha_1, \cdots, \cdots, \alpha_n\}$ and $\Sigma \vDash \beta$, then

$$\emptyset \vDash (\alpha_0 \to (\alpha_1 \to (\cdots \to (\alpha_n \to \beta) \cdots)).$$

Annotation 119 Deduction theorem 体现的淋漓尽致，将 β 完美转换成了一个 tautology.

Lemma 120 For any well-form formula γ containing atoms $p_1, p_2, \cdots, p_n$ and any valuation t, we have

1. If $\gamma^t = True$ then $\widehat{p}_1, \widehat{p}_2, \cdots, \widehat{p}_n \vdash \gamma$;
2. If $\gamma^t = False$ then $\widehat{p}_1, \widehat{p}_2, \cdots, \widehat{p}_n \vdash \neg\gamma$;

where defines $\widehat{p}_i$ as follow

$$\widehat{p}_i = \begin{cases} p_i & \text{if } p_i^t = True \\ \neg p_i & \text{if } p_i^t = False \end{cases}$$

Example 121 若 $\gamma = p \to q$，我们可以构造一个真值表

p	q	$p \to q$	Claim
T	T	T	$p, q \vdash p \to q$
T	F	F	$p, \neg q \vdash \neg(p \to q)$
F	T	T	$\neg p, q \vdash p \to q$
F	F	T	$\neg p, \neg q \vdash p \to q$

那么上面的 claims 是怎么来的呢？我们可以来分别证明, 对于第一行

$$\dfrac{\dfrac{\overline{p\ true}^{\ u} \quad q\ true}{q\ true}}{p \to q\ true}\ u$$

感觉有点奇怪，这里需要用到 vars inference rule, 这里相对于对 $q \vdash p \to q$ 的 weaken premise. 对于第二行

$$\dfrac{\dfrac{\overline{p \to q\ true}^{\ u} \quad p\ true}{q} \quad \neg q\ true}{\dfrac{\bot}{\neg(p \to q)\ true}\ u}$$

对于第三行

$$\dfrac{\dfrac{\overline{p\ true}^{\ u} \quad \neg p\ true}{\bot}}{p \to q\ true}\ u$$

对于第四行，和第三行类似. 可以看的出来这个 lemma 非常深刻，只要将 atoms 调整为在当前 valuation 下都是 true 的命题，结论再对应调整，就可以构造一个对应的 proof.

Lemma 122 For any well-formed formula γ, if $\emptyset \vDash \gamma$, then $\emptyset \vdash \gamma$.

Annotation 123 Lemma 122一句话概况就是 tautologies are provable. 其证明过程可以用 Lemma 120来说明. 现在 γ 是一个 tautology，那么对于所有的 valuation 都有 $\gamma^t = true$，这有什么用呢？这里还需要引入另外一种 tautology $p \vee \neg p$，配合 emilination rule of vee，即

$$\dfrac{(p_1 \vee \neg p_1) \quad (p_2 \vee \neg p_2) \quad \cdots \quad (p_n \vee \neg p_n) \qquad \begin{matrix}\overline{p_1} \ \cdots \ \overline{p_n} \\ \vdots \\ \gamma\end{matrix} \qquad \cdots \qquad \begin{matrix}\overline{\neg p_1} \ \cdots \ \overline{\neg p_n} \\ \vdots \\ \gamma\end{matrix}}{\gamma}$$

这里需要考虑有 2^n 个 cases，每一个对应一种 valuation，又因为 γ 是 tautology，因此最后的 conclusion 也都是 γ.

Lemma 124 If $\emptyset \vdash (\alpha_0 \to (\alpha_1 \to (\cdots \to (\alpha_n \to \beta)\cdots))$, then $\{\alpha_0, \alpha_1, \cdots, \alpha_n\} \vdash \beta$, that is, $\Sigma \vdash \beta$.

2.8 Derived Rules of Inference

Example 125

$$\frac{A \supset B \ true \quad B \supset C \ true}{A \supset C \ true}$$

is a derived rule of inference. Its derivation is the following:

$$\dfrac{\dfrac{B \supset C \ true \qquad \dfrac{A \supset B \ true \quad \overline{A \ true}^{\,u}}{B \ true} \supset E}{C \ true} \supset I^{u}}{A \supset C \ true}$$

Annotation 126 关于 derivation 的推导这里有一些 strategies 在里面

- 使用 introduction rule 从下至上，即我们想要什么;
- 使用 elimination rule 从上至下，即我们知道什么.

Example 127 Modus tollens(这玩意不就是逆否命题)

$$\frac{A \to B \quad \neg B}{\neg A} \ MT.$$

2.9 Curry-Howard Conrrespondence

Definition 128 Curry-Howard correspondence is between the natural deduction and simply-typed λ-calculus at three levels

- propositions are types;
- proofs are programs; and
- simplification of proofs is evaluation of programs.

That is

Types	Propositions
Unit types (1)	Truth ($\top$)
Product type ($\times$)	Conjunction ($\wedge$)
Union type ($+$)	Disjunction ($\vee$)
Function type ($\to$)	Implication ($\supset$)
Void types (0)	False ($\bot$)

Every typing rule has a correspondence with a deduction rule.

Example 129 The typing derivation of the term $\lambda a.\ \lambda b.\ \langle a, b\rangle$ can be seen as a deduction tree proving $A \supset B \supset A \wedge B$.

$$\dfrac{\dfrac{\dfrac{\dfrac{a : A \in \Gamma}{\Gamma \vdash a : A}\ var \quad \dfrac{b : B \in \Gamma}{\Gamma \vdash b : B}\ var}{\Gamma \vdash \langle a, b\rangle : A \times B}\ pair}{\Gamma \vdash \lambda y : B.\ \langle a, y\rangle : B \to A \times B}\ abs}{\Gamma \vdash \lambda x : A.\ \lambda y : B.\ \langle x, y\rangle : A \to B \to A \times B}\ abs \quad \Longleftrightarrow \quad \dfrac{\dfrac{\dfrac{\overline{A\ true}\ u \quad \overline{B\ true}\ w}{A \wedge B\ true}\ \wedge\wedge I}{B \supset A \wedge B\ true}\ \supset I^{w}}{A \supset B \supset A \wedge B\ true}\ \supset I^{u}$$

Annotation 130 从上面例子中看的出来，the inference rule of natural deduction 缺点什么，我也可以给原本每个 inference rule 都加上 the annotation for proof terms. [6] 那么这里 $M : A$ 有两种解释:

1. M is proof term for proposition A;
2. M is a program of type A.

这样解释 Curry-Howard ismorphism 或许方便一点. 让 proof terms make sense: 我们有"if $M : A$ then $A\ true$", 反过来"if $A\ true$ then $M : A$". 例如我们可以将 the proof term of $A \wedge B\ true$ 看做一个 pair 包含两个 subterm，一个关于 $A\ true$ 和另一个关于 $B\ true$.

$$\dfrac{M : A \quad N : B}{\langle M, N\rangle : A \wedge B}\ \wedge I$$

那么 the elimination rule of conjunction 对应一个 natural projection.

$$\frac{M : A \wedge B}{\pi_1 M : A}\ \wedge E_L \quad \frac{M : A \wedge B}{\pi_2 M : B}\ \wedge E_R$$

Example 131 通过 Curry-Howard isomorphism 我们可以将我们想要证明的 judgment 转换到 type system 中，你会看到非常的便利! 例如

$$(A \supset (B \wedge C)) \supset (A \supset B) \wedge (A \supset C)\ true$$

等价于

$$\lambda x.\ \langle \lambda y.\ \pi_1(x\ y), \lambda y.\ \pi_2(x\ y)\rangle : (A \to B \times C) \to (A \to B) \times (A \to C)$$

一个 implication 被转换成了对应的 abstraction，此时我们肯定会想如果给一个 false proposition 是不是就转不了？例如

$$(A \supset B) \supset (B \supset A)$$

显然我们无法在现有 type system 构造出一个合理的 abstraction 使得 $(A \to B) \to (B \to A)$.

迎面走来的问题是: 给定一个 proposition true, 是否有其他的 term with type 和它对应呢？显然是有的，

$$\lambda z.\ \lambda x.\ \langle \lambda y.\ \pi_1(x\ y), \lambda y.\ \pi_2(x\ y)\rangle\ z'$$

那这是不是违反 Curry-Howard isomorphsim 了呢？其实并不是，这里的对应是指 proof terms 和 deduction of proposition true, 显然 deduction 变了，对应的 proof terms 也要变.

Annotation 132 Curry-Howard isomorphism 建立在 ND 和 STLC 之间似乎看起来局限性很强，但是它是非常重要的第一步，是我们以另外一种视角来看待问题的方法，例如 well-type term is normalizing，那么也意味 well-formed formula is derivable. 之后 researcher 渐渐地发现了其他的 logics 也可以找到与之对应的 programming language, 直到现在这仍然是一个 active research field.

2.10 Quantifier

Annotation 133 natural deduction 很自然地建立在 propositional logic 上，现在要把 quantifier 加到里面，我们就得面向 first order logic 了，这里需要引入 terms 和 variables 来为后面 construction 做准备. 另外还得必要引入关于 term 的 properties，这里我们给它一个合适的名字叫做 type, 即 type of term, 为什么要引入这个它呢? 因为你可以想象我们经常在谈及 quantified statement 的时候，例如"every natural number is even or odd"，这里谈论的对象是 natural number，那么这个 natural number 实际上就是这里 term 的 type.

Definition 134 A formula $\forall x.A$ hold if and only if for every term a chosen, the proposition $A[a/x]$ hold, where the proposition means all occurrences of x in A replaced by a.

Definition 135 The introduction rule of universal quantifier is defined as follow:

$$\dfrac{\begin{array}{c}\overline{a:\tau}\\ \vdots \\ A[a/x]\ true\end{array}}{\forall x.A\ true}\ \forall I^a$$

where term a is called an *eigenvariable* and its shoule be fresh, meaning that it has not occurred anywhere else in the proof.

Definition 136 The emilination rule of universal quantifier is defined as follow:

$$\dfrac{\forall x.A\ true \quad t:\tau}{A[t/x]\ true}\ \forall E$$

where term t should not be any bound variables in A.

Definition 137 A formula $\exists x.A$ holds if and only if for some term t, $A[t/x]$ holds.

Definition 138 The introduction rule of existential quantifier is defined as follow:

$$\dfrac{A[t/x]\ true \quad t:\tau}{\exists x.A\ true}\ \exists I$$

where term t should not be any bound varaibles in A.

Definition 139 The emilination rule of existential quantifier is defined as follow:

$$\dfrac{\exists x.A\ true \qquad \begin{array}{c}\overline{a:\tau} \quad \overline{A[a/x]\ true}\\ \vdots \\ C\ true\end{array}}{C\ true}\ \exists E^{a,u}$$

where varaible a should be fresh.

Theorem 140 The rules for quantifier is local soundness and completeness

PROOF The local reduction and local expansion of universal quantifier as follow:

$$\dfrac{\dfrac{\overset{\overline{a:\tau}}{\overset{\mathcal{D}}{A[a/x]\ true}}}{\forall x.A\ true}\forall I^a \quad \overset{\mathcal{E}}{t:\tau}}{A[t/x]\ true}\forall E \qquad \Rightarrow_R \qquad \begin{array}{c}\overset{\mathcal{E}}{t:\tau}\\ \mathcal{D}[t/a]\\ A[t/x]\ true\end{array}$$

$$\overset{\mathcal{D}}{\forall x.A\ true} \qquad \Rightarrow_E \qquad \dfrac{\dfrac{\overset{\mathcal{D}}{\forall x.A\ true} \quad \overline{a:\tau}}{A[a/x]\ true}}{\forall x.A\ true}\forall I^a$$

The local reduction and local expansion of existential quantifier as follow:

$$\dfrac{\overset{\mathcal{D}}{A[t/x]\ true} \quad \overset{\mathcal{E}}{t:\tau}}{\exists x.A\ true}\exists I \qquad \dfrac{\overline{a:\tau} \quad \overline{A[a/x]\ true}^{\,u}}{\overset{\mathcal{F}}{C\ true}}}{C\ true}\exists E^{a,u} \qquad \Rightarrow_R \qquad \begin{array}{c}\overset{\mathcal{E}}{t:\tau} \quad \overset{\mathcal{D}}{A[a/x]\ true}\\ \mathcal{F}[t/a]\\ C\ true\end{array}$$

$$\overset{\mathcal{D}}{\exists x.A\ true} \qquad \Rightarrow_E \qquad \dfrac{\overset{\mathcal{D}}{\exists x.A\ true} \qquad \dfrac{\overline{a:\tau} \quad \overline{A[a/x]\ true}^{\,u}}{\exists x.A\ true}\exists I}{\exists x.A\ true}\exists E^{a,u}$$

Annotation 141 注意到 local expansion of existential quantifier 并不是以一个 introduction rule 结尾的，这和我预想的有些不太一样.

Annotation 142 Quantifiers corresponse dependent types.

3 Proof Searching

3.1 Natural Deduction in Sequent Nation

Definition 143 A sequent is a pariticular form of hypothetical judgement

$$A_1, \cdots, A_n \vdash C.$$

where $A_1, \cdots, A_n$ and C are well-defined formulas.

Definition 144 The correspondence between natural deduction and natural deduction in sequent nation.

$\dfrac{A\ true \quad B\ true}{A \wedge B\ true}\ \wedge I$	$\dfrac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \wedge B}\ \wedge I$
$\dfrac{A \wedge B\ true}{A\ true}\ \wedge E_1 \quad \dfrac{A \wedge B\ true}{B\ true}\ \wedge E_2$	$\dfrac{\Gamma \vdash A \wedge B}{\Gamma \vdash A}\ \wedge E_1 \quad \dfrac{\Gamma \vdash A \wedge B}{\Gamma \vdash B}\ \wedge E_2$
$\dfrac{A\ true}{A \vee B\ true}\ \vee I_1 \quad \dfrac{B\ true}{A \vee B\ true}\ \vee I_2$	$\dfrac{\Gamma \vdash A}{\Gamma \vdash A \vee B}\ \vee I_1 \quad \dfrac{\Gamma \vdash B}{\Gamma \vdash A \vee B}\ \vee I_2$
$\dfrac{A \vee B\ true \quad \begin{array}{c}\overline{A\ true}^{\,u} \\ \vdots \\ C\ true\end{array} \quad \begin{array}{c}\overline{B\ true}^{\,w} \\ \vdots \\ C\ true\end{array}}{C\ true}\ \vee E^{u,w}$	$\dfrac{\Gamma \vdash A \vee B \quad \Gamma, A \vdash C \quad \Gamma, B \vdash C}{\Gamma \vdash C}\ \vee E$
$\dfrac{\begin{array}{c}\overline{A\ true}^{\,u} \\ \vdots \\ B\ true\end{array}}{A \supset B\ true}\ \supset I^u$	$\dfrac{\Gamma, A \vdash B}{\Gamma \vdash A \supset B}\ \supset I$
$\dfrac{A \supset B\ true \quad A\ true}{B\ true}\ \supset E$	$\dfrac{\Gamma \vdash A \supset B \quad \Gamma \vdash A}{\Gamma \vdash B}\ \supset E$
$\dfrac{}{\top\ true}\ \top I$	$\dfrac{}{\Gamma \vdash \top}\ \top I$
$\dfrac{\bot\ true}{C\ true}\ \bot E$	$\dfrac{\Gamma \vdash \bot}{\Gamma \vdash C}\ \bot E$
Hypothesis discharging	$\dfrac{}{\Gamma, A \vdash A}\ hyp$
Substitution	$\dfrac{\Gamma, A \vdash C \quad \Gamma \vdash A}{\Gamma \vdash C}\ subst$

Annotation 145 (detail of correspondence) 其中 Γ 是一个 set of formulas，它可以是 empty set. 思考上述 sequent 形式下的 natural deduction，我们应该用 bottom-up 的视角来观察. 试想我们在没有 additional assumptions 证明一个 formulas，在最开始 Γ 应该是 empty 的，随着我们不断 apply 上述规则过程中将不断的填充 Γ. 那么什么时候证明算接结束了呢? 在 natural deduction 中我们从下往上使用 introduction rules，并添加相应的 assumptions，再从上往下使用 emilination rules，直到它们在中途相遇，这时候我们的证明就结束了，当证明结束的时候，此时所有的 assumptions 都应该被 discharge 了，这个操作对应到 sequent 形式下就是上述 hyp rule，在利用 sequent 构造 proof 的时候，总是以 hyp rule 结束的.

显然 sequent 提供了一种更加优雅的收集 assumptions 的方式，我们总是将 new assumption 放在 left side，使得 assumptions 和需要证明的 goal formula 总是在一个 level. 一旦某个 assumption 建立之后，以此往后的证明过程中这个 assumption 都是 visible 的，仔细想想这是对的，在 natural deduction 中，我们总是在最上面添加新的 assumption.

Example 146 A proof in sequent form.

$$\dfrac{\dfrac{\dfrac{}{A\supset B,(A\wedge C)\vdash A\supset B}\,hyp \quad \dfrac{\dfrac{}{A\supset B,(A\wedge C)\vdash A\wedge C}\,hyp}{A\supset B,(A\wedge C)\vdash A}\,\wedge E_1}{A\supset B,(A\wedge C)\vdash B}\,\supset E \quad \dfrac{\dfrac{}{A\supset B,(A\wedge C)\vdash A\wedge C}\,hyp}{A\supset B,(A\wedge C)\vdash C}\,\wedge E_2}{\dfrac{A\supset B,(A\wedge C)\vdash (B\wedge C)}{A\supset B\vdash (A\wedge C)\supset (B\wedge C)}\,\supset I}{\vdash (A\supset B)\supset ((A\wedge C)\supset (B\wedge C))}\,\supset I$$

Annotation 147 最后提一下最后 *subst* rule，在 ND 中 substitution 88是指我们可以将某个 assumption 替换成关于它的一个 derivation. *subset* rule 也是一个意思，但是你会发现 premise 和 conclusion 都使用了相同 assumpations(contexts) Γ, 这是因为下面的 weakning lemma 在这里做了保证.

Lemma 148 If $F\vdash C$, then $\Gamma, F\vdash C$.

Definition 149 We say a rule is admissible if all proofs using the rule can be transformed into proofs that do not use the rule.

Lemma 150 The *subst* rule is admissible.

Proof 我们可以将其替换为等价的形式

$$\dfrac{\Gamma,A\vdash C \quad \Gamma\vdash A}{\Gamma\vdash C}\,subst \quad \rightsquigarrow \quad \dfrac{\dfrac{\Gamma,A\vdash C}{\Gamma\vdash A\supset C}\supset I \quad \Gamma\vdash A}{\Gamma\vdash C}\supset E$$

3.2 Sequent Calculus

Annotation 151 在前一章里面我们把 ND 转化成了 ND in sequent nation，这一章我们就来把 verification calculus 转化为 sequent calculus. 为了更好的说明 verification calculus 和 sequent calculus 的对应，我们下面在定义 sequent calculus 的时候，会象征性地加上 $left$ 和 $right$ 对应 use 和 verification. 准确地说这里应该是 intuitionistic sequent calculus, 每个 sequent 的 left side 只有一个 formula

Definition 152 A sequent is a pariticular form of hypothetical judgement

$$A_1\ left, \cdots, A_n\ left \vdash C\ right.$$

where $A\ left$ corresponds to a proposition that can be used ($A \downarrow$) and $C\ right$ corresponds to a proposition we have to verify ($C \uparrow$). The right rules decompose C in analogy with introduction rules from the perspective of "bottom-up", while the left rule decompose one of the hypotheses, in analogy with elimination rules, but "upside-down".

Definition 153 The initial rule

$$\frac{}{\Gamma, P\ left \vdash P\ right}\ init$$

where P is atomic proposition.

Definition 154 The left rules and right rules

$\dfrac{\Gamma, A \wedge B\ left, A\ left \vdash C\ right}{\Gamma, A \wedge B\ left \vdash C\ right}\ \wedge L_1$ $\dfrac{\Gamma, A \wedge B\ left, B\ left \vdash C\ right}{\Gamma, A \wedge B\ left \vdash C\ right}\ \wedge L_2$	$\dfrac{\Gamma \vdash A\ right \quad \Gamma \vdash B\ right}{\Gamma \vdash A \wedge B\ right}\ \wedge R$
$\dfrac{\Gamma, A \supset B\ left \vdash A\ right \quad \Gamma, A \supset B\ left, B\ left \vdash C\ right}{\Gamma, A \supset B\ left \vdash C\ right}\ \supset L$	$\dfrac{\Gamma, A\ left \vdash B\ right}{\Gamma \vdash A\ left \supset B\ right}\ \supset R$
$\dfrac{\Gamma, A \vee B\ left, A\ left \vdash C\ right \quad \Gamma, A \vee B\ left, B\ left \vdash C\ right}{\Gamma, A \vee B\ left \vdash C\ right}\ \vee L$	$\dfrac{A\ right}{\Gamma \vdash A \vee B\ right}\ \vee R_1$ $\dfrac{B\ right}{\Gamma \vdash A \vee B\ right}\ \vee R_2$
	$\dfrac{}{\Gamma \vdash \top\ right}\ \top R$
$\dfrac{}{\Gamma, \bot\ left \vdash C\ right}\ \bot L$	

Annotation 155 The above rules we can use $\Gamma \Rightarrow A$ instead of them.

Annotation 156 这里 frank 给出的 left rules 怪怪的，因为 conclusion 里面的 assumptions 依然出现在了 primises 里面，这让人很奇怪，虽然不影响其正确性. frank 对此的意见是这只是一种 weakening 操作，同时他想表达一个"monotonicity of hypotheses" 的概念: 在 bottom-up 形式下的 proof 中一旦建立某个 assumption，那么它在后续的构造过程中同样 available.

我的感觉是 left rules 应该和 right rules 一样，right rules 在 simplify conclusion，而 left rules 也应该去 simplify hypotheses. 这里 simplify 是指去掉 formula 里面存在的 connectives.

Example 157 The proof in sequent calculus.

$$\dfrac{\dfrac{\dfrac{\dfrac{}{A \supset B, (A \wedge C), A \Rightarrow A}\, init}{A \supset B, (A \wedge C) \Rightarrow A}\, \wedge L_1 \quad \dfrac{}{A \supset B, (A \wedge C), B \Rightarrow B}\, init}{A \supset B, (A \wedge C) \Rightarrow B}\, \supset L \quad \dfrac{\dfrac{}{A \supset B, (A \wedge C), C \Rightarrow C}\, init}{A \supset B, (A \wedge C) \Rightarrow C}\, \wedge L_2}{A \supset B, (A \wedge C) \Rightarrow (B \wedge C)}\, \wedge R$$
$$\dfrac{\dfrac{A \supset B, (A \wedge C) \Rightarrow (B \wedge C)}{A \supset B \Rightarrow (A \wedge C) \supset (B \wedge C)}\, \supset R}{\Rightarrow (A \supset B) \supset ((A \wedge C) \supset (B \wedge C))}\, \supset R$$

Theorem 158 (from verifications to sequent calculus) Given hypotheses $\Gamma = (A_1 \uparrow, \cdots, A_n \uparrow)$, it correpsonds to $\widehat{\Gamma} = (A_1\ left, \cdots, A_n\ left)$. Then we have

1. If $\Gamma \vdash C \uparrow$ then $\widehat{\Gamma} \vdash C\ right$;
2. If $\Gamma \vdash A \downarrow$ and $\widehat{\Gamma}, A\ left \vdash C\ right$ then $\widehat{\Gamma} \vdash C\ right$.

Proof 这里需要对 $\Gamma \vdash C \uparrow$ 和 $\Gamma \vdash A \downarrow$ 做 mutual induction. 记录几个 representative cases.

Case 1 若

$$\dfrac{\Gamma, C_1 \downarrow \vdash C_2 \uparrow}{\Gamma \vdash C_1 \supset C_2 \uparrow}\, \supset I$$

则

(1) $\widehat{\Gamma}, C_1\ left \vdash C_2\ right$ hyp.1 from premise1

(2) $\widehat{\Gamma} \vdash C_1 \supset C_2\ right$ $\supset R$. (1)

Case 2 若

$$\dfrac{\Gamma \vdash P \downarrow}{\Gamma \vdash P \uparrow}\, \downarrow\uparrow$$

则

(1) $\widehat{\Gamma}, P\ left \vdash P\ right$ *init*

(2) $\widehat{\Gamma} \vdash P\ right$ hyp.2 from premise1

Case 3 若

$$\frac{\Gamma \vdash A_1 \supset A_2 \downarrow \quad \Gamma \vdash A_1 \uparrow}{\Gamma \vdash A_2 \downarrow} \supset E$$

则

(1)	$\widehat{\Gamma}, A_2\ left \vdash C\ right$	assumption
(2)	$\widehat{\Gamma}, A_1 \supset A_2\ left, A_2\ left \vdash C\ right$	weakening(1)
(3)	$\widehat{\Gamma} \vdash A_2\ right$	hyp.1 from premise1
(4)	$\widehat{\Gamma}, A_1 \supset A_2\ left \vdash A_2\ right$	weakening(3)
(5)	$\widehat{\Gamma}, A_1 \supset A_2\ left \vdash C\ right$	$\supset L(2)(4)$
(6)	$\widehat{\Gamma} \vdash C\ right$	hyp.1 from premise2

Case 4 若

$$\frac{}{\Gamma', A \downarrow \vdash A \downarrow}\ hyp$$

则

(1)	$\widehat{\Gamma}, A\ left, A\ left \vdash C\ right$	assumption
(2)	$\widehat{\Gamma}, A\ left \vdash C\ right$	contraction(1)

Annotation 159 注意这里的 hypotheses 是两个部分，因为 verification calculus 的 elimintation rule 存在，会导致 use 同样出现在左端. 实际这里缺一个过程，应该像 ND in Seq 那样，我们也应该对 verificatio calculus 也做一个 sequent 形式的变换.

Theorem 160 (substitution of uses) If $\Gamma \vdash A \downarrow$ then

1. if $\Gamma, A \downarrow \vdash B \downarrow$ then $\Gamma \vdash B \downarrow$, and
2. if $\Gamma, A \downarrow \vdash C \uparrow$ then $\Gamma \vdash C \uparrow$,

Proof 这里需要对 $\Gamma, A \downarrow \vdash B \downarrow$ 和 $\Gamma, A \downarrow \vdash C \uparrow$ 做 mutual induction. 还是列举几个代表性的 cases.

Case 1 Base case

$$\frac{}{\Gamma \vdash \top \uparrow}\ \top I$$

根据假设 $\Gamma, A \downarrow \vdash \top \uparrow$，这里显然有 $\Gamma, \vdash \top \uparrow$.

Case 2 若

$$\frac{\Gamma' \vdash C \supset B \downarrow \quad \Gamma' \vdash C \uparrow}{\Gamma' \vdash B \downarrow} \supset E$$

其中 $\Gamma' = (\Gamma, A\downarrow)$. 那么

$$\begin{array}{lll} (1) & \Gamma \vdash A\downarrow & \text{assumption} \\ (2) & \Gamma \vdash C \supset B\downarrow & \text{hyp.1} \\ (3) & \Gamma \vdash C\uparrow & \text{hyp.2} \\ (4) & \Gamma \vdash B\downarrow & \supset E(3)(4) \end{array}$$

Theorem 161 (from sequent calculus to verifications) If $\widehat{\Gamma} \vdash C\ right$ then $\Gamma \vdash C\uparrow$.

Proof 注意这里有个 abuse symbol 了，结合前面的 theorem，可能会想成我们构造了一个 isomorphism，其实不是这样的，我仅仅讨论从一边到另一边，并不是 composition! 这里依然对 $\widehat{\Gamma} \vdash C\ right$ 做 structure induction. 列举几个代表性 cases.

Case 1 Base case

$$\frac{}{\widehat{\Gamma} \vdash \top\ right}\ \top R$$

显然有 $\Gamma \vdash \top\uparrow$.

Case 2 若

$$\frac{\widehat{\Gamma}, A\ left \vdash B\ right}{\widehat{\Gamma} \vdash A \supset B\ right}\ \supset R$$

则

$$\begin{array}{lll} (1) & \Gamma, A\uparrow \vdash B\uparrow & \text{hyp} \\ (2) & \Gamma \vdash A \supset B\uparrow & \supset I(1) \end{array}$$

Case 3 若

$$\frac{\widehat{\Gamma}, A \supset B\ left \vdash A\ right \quad \widehat{\Gamma}, A \supset B\ left, B\ left \vdash C\ right}{\widehat{\Gamma}, A \supset B\ left \vdash C\ right}\ \supset L$$

则

$$\begin{array}{lll} (1) & \Gamma, A \supset B\downarrow \vdash A\uparrow & \text{hyp} \\ (2) & \Gamma, A \supset B\downarrow \vdash A \supset B\downarrow & \textit{hyp}\ \text{rule} \\ (3) & \Gamma, A \supset B\downarrow \vdash B\downarrow & \supset E(1)(2) \\ (4) & \Gamma, A \supset B\downarrow, B\downarrow \vdash C\uparrow & \text{hyp} \\ (n) & \Gamma, A \supset B \vdash C\uparrow & \textit{subst}(160) \end{array}$$

Definition 162 (another of substitution) The rule of cut

$$\frac{\Gamma \vdash A\ right \quad \Gamma, A\ left \vdash C\ right}{\Gamma \vdash C\ right}\ cut$$

Annotation 163 注意 *cut* rule 是在用 the vertification of A 去替换 the use of A，这和前面 substitution of uses 是不太一样的.

Theorem 164 (admissibility of cut) If $\Gamma \vdash A\ right$ and $\Gamma, A\ left \vdash C\ right$ then $\Gamma \vdash C\ right$.

Proof 证明 *cut* rule 是个技术活. 我们要做一个 nested structure induction. 首先给定 *cut* rule 的 shape

$$\frac{\overset{\mathcal{D}}{\Gamma \Rightarrow A} \quad \overset{\mathcal{E}}{\Gamma, A \Rightarrow C}}{\Gamma \Rightarrow C}\ cut$$

这里我们要对 triple $(C, \mathcal{D}, \mathcal{E})$ 做归纳. 有三种 base cases:

1. 若 $\mathcal{E}$ 是 *init* rule. 这里两个地方可以 apply *init* rule. 此时 A 为 atomic.

   (a) 若 A 不同于 C. 那么这里可以马上知道 $A \in \Gamma$，因此可以 emilinate 掉 cut.

   $$\frac{\overset{\mathcal{D}}{\Gamma \Rightarrow C} \quad \dfrac{}{\Gamma, C \Rightarrow A}\ init}{\Gamma \Rightarrow A}\ cut \rightsquigarrow \frac{}{\Gamma \Rightarrow A}\ init$$

   (b) 若 $C = A$. 那么这里显然可以直接用 $\mathcal{D}$ 得到 $\Gamma \Rightarrow A$.

   $$\frac{\overset{\mathcal{D}}{\Gamma \Rightarrow A} \quad \dfrac{}{\Gamma, A \Rightarrow A}\ init}{\Gamma \Rightarrow A}\ cut \rightsquigarrow \overset{\mathcal{D}}{\Gamma \Rightarrow A}$$

2. 若 $\mathcal{D}$ 是 *init* rule. 此时 C 是 atomic，那么可以知道 $C \in \Gamma$，因此 $\Gamma', C, C \Rightarrow A$，再用一下 contraction 就有 $\Gamma \Rightarrow A$，因此这里可以用 $\mathcal{E}$ 作为 proof derivation.

   $$\frac{\dfrac{}{\Gamma \Rightarrow C}\ init \quad \overset{\mathcal{E}}{\Gamma, C \Rightarrow A}}{\Gamma \Rightarrow A}\ cut \rightsquigarrow \overset{\mathcal{E}}{\Gamma \Rightarrow A}.$$

3. 若 C 是 atomic. 此时有可能 $\mathcal{D}$ 和 $\mathcal{E}$ 都不是 *init* rule, 此时我们需要一个 lemma 167 来使得它们变成前面两种情况.

那么我们怎么来用这三个 base case，这里就展示一下 nested induction 是咋 worked. 首先我们给定一个命题 $cut(F, l, r)$: 如果 l 是一个关于 $\Gamma \Rightarrow F$ 的 cut-free proof，r 是一个关于 $\Gamma, F \Rightarrow C$ 的 cut-free proof，那么我们可以构造一个关于 $\Gamma \Rightarrow C$ 的 cut-free proof. 来正式开始我的 induction.

- [11]Base case(1): $\forall l.\forall r.cut(atom, l, r)$

  对应前面的 base case(3).

- Induction step(1): to show $\forall l.\forall r.cut(F+1, l, r)$

  IH(1): $\forall l.\forall r.cut(F, l, r)$

  The proof procceeds by induction on l:

  - Base case(2): $\forall l.\forall r.cut(F+1, init, r)$ 对应前面的 base case(2).
  - Induction step(2): to show $\forall l.\forall r.cut(F+1, l+1, r)$

    IH(2): $\forall l.\forall r.cut(F+1, l, r)$

    The proof procceeds by induction on r:

    1. Base case(3): $\forall l.\forall r.cut(F+1, l+1, init)$
    2. Induction step(3): to show $\forall l.\forall r.cut(F+1, l+1, r+1)$

       IH(3): $\forall l.\forall r.cut(F+1, l+1, r)$

       Now we show this by cases. As a bleow example:

$$\dfrac{\dfrac{\overset{\mathcal{D}}{\Gamma \Rightarrow A} \quad \overset{\mathcal{E}}{\Gamma \Rightarrow B}}{\Gamma \Rightarrow A \wedge B} \wedge L_1 \quad \dfrac{\overset{\mathcal{F}}{\Gamma, A \wedge B, A \Rightarrow C}}{\Gamma, A \wedge B \Rightarrow C}}{\Gamma \Rightarrow C}\, cut$$

$$\Downarrow$$

$$\dfrac{\overset{\mathcal{D}}{\Gamma \Rightarrow A} \qquad \dfrac{\dfrac{\mathcal{D} + \text{ weakening}}{\Gamma, A \Rightarrow A} \quad \dfrac{\mathcal{E} + \text{ weakening}}{\Gamma, A \Rightarrow B}}{\Gamma, A \Rightarrow A \wedge B} \wedge R \quad \dfrac{\overset{\mathcal{F}}{\Gamma, A \wedge B, A \Rightarrow C}}{\Gamma, A \Rightarrow C}\, cut}{\Gamma \Rightarrow C}\, cut$$

       on the upper-most cut, we apply IH(3), since $F+1$ and $l+1$ are unchanged, but right branch is now r, which is smaller. and on the lower cut, we apply IH(1), since we have F as the cut-formula. □

Definition 165 We call rank deduction the rewriting operation that premutes the cut rule over other rules in a proof.

Annotation 166 "premutes it up" 是啥意思呢? 就是指某个 cut rule" 往上移", 例如

$$\dfrac{\dfrac{\dfrac{}{\Gamma, A \wedge B, A \Rightarrow A}\ init}{\Gamma, A \wedge B \Rightarrow A}\ \wedge L_1 \qquad \overset{\mathcal{E}}{\Gamma, A \wedge B, A \Rightarrow C}}{\Gamma, A \wedge B \Rightarrow C}\ cut$$

$$\Downarrow$$

$$\dfrac{\dfrac{}{\Gamma, A \wedge B, A \Rightarrow A}\ init \qquad \begin{array}{c}\mathcal{E} + \text{ weakening} \\ \Gamma, A \wedge B, A, A \Rightarrow C\end{array}}{\dfrac{F, A \wedge B, A \Rightarrow C}{\Gamma, A \wedge B \Rightarrow C}\ \wedge L_1}\ cut$$

此时 cup 拥有了 smaller left branch, 这其中为我们后续的 nested induction hypothesis 提供了途径. 再来看一个例子

$$\dfrac{\dfrac{\overset{\mathcal{D}}{\Gamma \Rightarrow A} \qquad \overset{\mathcal{E}}{\Gamma \Rightarrow B}}{\Gamma \Rightarrow A \wedge B} \qquad \dfrac{\overset{\mathcal{F}}{\Gamma, A \wedge B, A \Rightarrow C}}{\Gamma, A \wedge B \Rightarrow C}\ \wedge L_1}{\Gamma \Rightarrow C}\ cut$$

$$\Downarrow$$

$$\dfrac{\overset{\mathcal{D}}{\Gamma \Rightarrow A} \qquad \dfrac{\dfrac{\begin{array}{c}\mathcal{D} + \text{ weakening} \\ \Gamma, A \Rightarrow A\end{array} \qquad \begin{array}{c}\mathcal{E} + \text{ weakening} \\ \Gamma, A \Rightarrow B\end{array}}{\Gamma, A \Rightarrow A \wedge B}\ \wedge R \qquad \overset{\mathcal{F}}{\Gamma, A \wedge B, A \Rightarrow C}}{\Gamma, A \Rightarrow C}\ cut}{\Gamma \Rightarrow C}\ cut$$

此时 *cut rule* 中的两个 premises 中的 cut formula 都是被 apply 了相应的 rules, 这种 shape 的 cut 我称为principal cases, 它和第一个例子不太相同. 此时下面这个 cup 拥有了 smaller left branch, 而上面这个 cup 拥有了 smaller right branch.

Lemma 167 The *cut* rule permutes up all other rules that do not operate on the cut formula.

Proof 换句话说就是对于任意的 *cut* rule, 我们都可以重排它的 primises, 把 cut 放到 right place. 那么这里分别要对它的 primises 应用 sequent calculus 的规则, 两个 primises 就是 20cases. td □

Annotation 168 引入 *cut* rule 不利于做 proof search.

Definition 169 (generalization of *init* rule) The rule of identity

$$\dfrac{}{\Gamma, A\ left \vdash A\ right}\ id$$

Theorem 170 (admissibility of identity) $\Gamma, A\ left \vdash A\ right$ fir arbitrary propositions A and contexts Γ.

PROOF 对 A 做 structure induction. 列举几个代表性的 cases.

Case 1 若 $A = P$, 根据 *init* rule 显然有 $\Gamma, A\ left \vdash A\ right$.

Case 2 若 $A = B \supset C$. 则

(1)	$\Gamma, B\ left \vdash B\ right$	hyp
(2)	$\Gamma, B \supset C\ left, B\ left \vdash B\ right$	weakening
(3)	$\Gamma, C\ left \vdash C\ right$	hyp
(4)	$\Gamma, B \supset C\ left, B\ left, C\ left \vdash C\ right$	weakning
(5)	$\Gamma, B \supset C\ left, B\ left \vdash C\ right$	$\supset L(2)(4)$
(6)	$\Gamma, B \supset C\ left \vdash B \supset C\ right$	$\supset R(5)$

Case 3 若 $A = B \wedge C$. 则

(1)	$\Gamma, B \wedge B, B \Rightarrow B$	hyp + weakening
(2)	$\Gamma, B \wedge C \Rightarrow C$	$\wedge L_1(1)$
(3)	$\Gamma, B \wedge C, C \Rightarrow C$	hyp + weakening
(4)	$\Gamma, B \wedge C \Rightarrow B$	$\wedge L_2(3)$
(5)	$\Gamma, B \wedge C \Rightarrow B \wedge C$	$\wedge R(2)(4)$

Theorem 171 (from natural deduction to sequent calculus) If $\Gamma \vdash A\ true$ then $\widehat{\Gamma} \vdash A\ right$.

PROOF 依然对 $\Gamma \vdash A\ true$ 做 structure induction. 列举几种代表性 cases.

Case 1 若

$$\frac{A\ true}{\Gamma', A\ true}\ hyp$$

此时 $\Gamma = (\Gamma', A\ true)$, 根据 *identity* rule(169) 有 $\widehat{\Gamma}, A\ left \vdash A\ right$.

Case 2 若

$$\frac{\Gamma, B\ true \vdash C\ true}{\Gamma \vdash B \supset C\ true} \supset I$$

则

(1)	$\widehat{\Gamma}, A\ left \vdash B\ right$	hyp
(2)	$\widehat{\Gamma} \vdash A \supset B\ right$	$\supset R(1)$

Case 3 若

$$\frac{\Gamma \vdash C \supset B\ true \quad \Gamma \vdash C\ true}{\Gamma \vdash B\ true} \supset E$$

则

(1)	$\widehat{\Gamma} \vdash C \supset B\ right$	hyp
(2)	$\widehat{\Gamma} \vdash C\ right$	hyp
(3)	$\widehat{\Gamma}, C \supset B\ left, C\ left \vdash C\ right$	*identity*
(4)	$\widehat{\Gamma}, C \supset B\ left, C\ left, B\ left \vdash B\ right$	*identity*
(5)	$\widehat{\Gamma}, C \supset B\ left, C\ left \vdash B\ right$	$\supset L(3)(4)$
(6)	$\widehat{\Gamma}, C\ left \vdash B\ right$	$cut(1)(5)$
(7)	$\widehat{\Gamma} \vdash B\ right$	$cut(2)(6)$

Theorem 172 (truth and verification) $A\ true$ iff $A\uparrow$.

Proof (⇒) 从 $A\updownarrow$ 到 $A\ true$ 是比较显然的，直接将所有的 arrow 都换成 true 即可，对于 arrow switch $\downarrow\uparrow$，此时 premise 和 conclusion 都是相同，因此这里可以在转换中去掉.

(⇐) 这里就需要多出中间一步. 利用 Theorem 171 将 $\cdot \vdash A\ true$ 转换到 $\cdot \vdash A\ right$; 再利用 Theorem 161 将 $\cdot \vdash A\ right$ 转换到 $\cdot \vdash A\ \uparrow$. □

Annotation 173 How everything is related

Natural Deduction

$\equiv$

ND in sequent nation

$\Updownarrow$

Sequent Calculus(with cut, identity and weakening)

$\updownarrow$

Sequent Calculus

$\equiv$

Verification Calculus

where $\equiv$ means the systems are equivalent, $\Updownarrow$ means the systems are sound and complete each other, and $\updownarrow$ means the system are shown to be equivalent by showing the rules cut, identity and weakening are admissible.

3.3 Simplification

Annotation 174 目的是去掉 sequent calculus 里面 the duplication of main formula. 为了区分 simplified sequent，我们用 $\to$ 代替 $\Rightarrow$

Definition 175 [12]Simplified sequent calculus defined as follow

$\dfrac{\Gamma, A, B \to C}{\Gamma, A \wedge B \to C} \wedge L$	$\dfrac{\Gamma \to A \quad \Gamma \to B}{\Gamma \to A \wedge B} \wedge R$
$\dfrac{\Gamma, A \to C \quad \Gamma, B \to C}{\Gamma, A \vee B \to C} \vee L$	$\dfrac{A}{\Gamma \to A \vee B} \vee R_1$ $\dfrac{B}{\Gamma \to A \vee B} \vee R_2$
$\dfrac{\Gamma, A \supset B \to A \quad \Gamma, B \to C}{\Gamma, A \supset B \to C} \supset L$	$\dfrac{\Gamma, A \to B}{\Gamma \to A \supset B} \supset R$
	$\dfrac{}{\Gamma \to \top} \top R$
$\dfrac{}{\Gamma, \bot \to C} \bot L$	
$\dfrac{}{\Gamma, P \to P}$	

Annotation 176 为什么 implication emilination 的 left premise 中的 main formual 没有去掉? 这是值得探讨的问题，right premise 能去掉因为

$$\dfrac{\dfrac{}{\Gamma, A \supset B \Rightarrow A \supset B}\ id \qquad \Gamma, A \supset B, B \vdash C}{\Gamma, B \Rightarrow C}\ cut$$

同时更加底层的原因是 $\vdash B \supset (A \supset B)$. 而你此时思考 $\Gamma, A \supset B \supset A$ 时，应当思考是否存在一个关于 A 的 proof 里面需要用到 $A \supset B$? 注意此时的 *cut* rule 在这里是无法奏效的，我好想想不到这样 proof td.

Theorem 177 Simplified sequent calculus is soundness and completeness, that is

- If $\Gamma \to C$ then $\Gamma \Rightarrow C$;
- If $\Gamma \Rightarrow C$ then $\Gamma \to C$.

Proof 依旧是 straightforward structure induction. □

3.4 Invertibility

Annotation 178 Why Invertibility? [13] 我们想构造一个 automation for proof searching，可能会面临一些问题. 假设我们想要找 $\Gamma \vdash C$ 的 verification，其中 Γ 包含 $n-1$ 个 formuals，这意味我们要从 n 个 formuals(加上 C)，挑一个出来作为 main formual，对其 apply 对应的 rules. 如果我们挑到正确的那个 rule，将会导致都后续一定会出错，因此我们要考虑 backtrack. 例如证明 $B \to A \vee B$ 时，我们对 right side 挑 $\vee R_1$ rule

$$\frac{\Gamma \to A}{\Gamma \to A \vee B} \vee R_1,$$

这显然就出问题了. 这样看起来我们得找到一个可以接受的 search algorithm，这里有策略是我们可以尽可能选 right choice，直到选不出时候，我们再 make a choice. 这样做的好处是我们可以尽可能减少 backtrack，同时一旦选择了 bad choice，那么无论你后续怎么选都还是错的，因此我还可以直接 remove 掉 bad choice. 那么如何选 right choice，就引出了 invertible rule 的概念.

Definition 179 A rule p is called invertible in a sequent calculus system if a proof of its conclusion implies the existence of proofs of each of its premises.

Annotation 180 简而言之就是可以从 conclusion is vaild 推出 premise is vaild.

Lemma 181 The left rule for disjunction is invertible.

$$\frac{\Gamma, A \to C \quad \Gamma, B \to C}{\Gamma, A \vee B \to C} \vee L$$

Proof 很直接.

$$\frac{\dfrac{\dfrac{}{\Gamma, A \to A}\, id}{\Gamma, A \to A \vee B} \vee R_1 \quad \Gamma, A, A \vee B \to C}{\Gamma, A \to C}\, cut \qquad \frac{\dfrac{\dfrac{}{\Gamma, B \to B}\, id}{\Gamma, B \to A \vee B} \vee R_2 \quad \Gamma, B, A \vee B \to C}{\Gamma, B \to C}\, cut$$

Lemma 182 Both rules for conjunction are invertible.

Lemma 183 The right rule for implication is invertible.

Proof that is

$$\frac{\Gamma, A \to A \supset B \qquad \dfrac{\dfrac{}{\Gamma, A, A \supset B \to A}\, init \quad \dfrac{}{\Gamma, A, B \to B}\, init}{\Gamma, A, A \supset B \to B}}{\Gamma, A \to B}$$

Annotation 184 为什么 the left rule of implication is not invertible? 因为从 conclusion 推不出 left premise.

Definition 185 Proof searching algorithm. Assuming we want prove $\Gamma \to C$

1. Starting with formula on right C, apply invertible rules as long as we can
2. When the only rule left to be applied on the right is non-invertible, we say C^+, stop working there and move to left.
3. Process the formulas in the left context Ω in order:
   (a) If the rule to be applied is invertible, go ahead and apply it, keeping the possibly generated subformuals in the front of the list.
   (b) If the rule to be applied is non-invertible or the formula is atomic, move it to a side context Γ^- to delay working with it.
4. When we have absolutely no other choice, we can either apply a noninvertible rule on the right or on the left. Then we move the focus to the newly generated formulas in the hope they require inveritible.

that Ω is context of left side, which is ordered and holds any formuals; and Γ^- which holds only atoms or formuals that require non-invertible left rules.

Annotation 186 接下来就是详细解释 Ω 和 Γ^- 是如何构造的.

Definition 187 We will label the sequent arrows with R or L, depending on whether we are on a right inversion phase or left inversion phase.

$$\Gamma^-; \Omega \xrightarrow{R} C$$
$$\Gamma^-; \Omega \xrightarrow{L} C$$

Definition 188 Formalization of proof searching algorithm for $\Gamma \to C$.

1. Start: $\cdot; \Omega \xrightarrow{R} C$, where Γ^- is empty and $\Omega = \Gamma$.
2. Process right side:
   (a) The right inversion phase consists of applying right invertibale rule:

$$\frac{\Gamma^-; \Omega \xrightarrow{R} A \quad \Gamma^-; \Omega \xrightarrow{R} B}{\Gamma^-; \Omega \xrightarrow{R} A \wedge B} \wedge R$$
$$\frac{\Gamma^-; \Omega, A \xrightarrow{R} B}{\Gamma^-; \Omega \xrightarrow{R} A \supset B} \supset R$$
$$\frac{}{\Gamma^-; \Omega \xrightarrow{R} \top} \top R$$

(b) If we reach an atom on the right side, we either check if it is in Γ^- and close the branch with *init*, or we move to apply left inversion rules.

$$\frac{P \in \Gamma^-}{\Gamma^-; \Omega \xrightarrow{R} P}\ \mathit{init}$$

$$\frac{P \notin \Gamma^- \quad \Gamma^-; \Omega \xrightarrow{L} P}{\Gamma^-; \Omega \xrightarrow{R} P}\ \mathrm{LR}_P$$

(c) The only other cases left are when the right formula is disjunction or $\bot$. At this pointwe stop working on the right and move to the left.

$$\frac{\Gamma^-; \Omega \xrightarrow{L} A \vee B}{\Gamma^-; \Omega \xrightarrow{R} A \vee B}\ \mathrm{LR}_\vee$$

$$\frac{\Gamma^-; \Omega \xrightarrow{L} \bot}{\Gamma^-; \Omega \xrightarrow{R} \bot}\ \mathrm{LR}_\bot$$

3. Process left side:

(a) The left inversion phase processes the formulas in Ω in order, always taking the rightmost one.

$$\frac{\Gamma^-; \Omega, A, B \xrightarrow{L} C^+}{\Gamma^-; \Omega, A \wedge B \xrightarrow{L} C^+}\ \wedge L$$

$$\frac{\Gamma^-; \Omega, A \xrightarrow{L} C^+ \quad \Gamma^-; \Omega, B \xrightarrow{L} C^+}{\Gamma^-; \Omega, A \vee B \xrightarrow{L} C^+}\ \vee L$$

(b) If the first formual in Ω is $\bot$, we can close the branch; If it is $\top$ we can remove it form our list.

$$\frac{}{\Gamma^-; \Omega, \bot \xrightarrow{L} C^+}\ \bot\, L$$

$$\frac{\Gamma^-; \Omega \xrightarrow{L} C^+}{\Gamma^-; \Omega, \top \xrightarrow{L} C^+}\ \top L$$

(c) If we encounter an atom, we can close the branch if it is equal to the right side or we can move it. and only case left is an implication, we also can move it to Γ^-.

$$\frac{}{\Gamma^-; \Omega, P \xrightarrow{L} P}\ \mathit{init}$$

$$\frac{\Gamma^-, P; \Omega \xrightarrow{L} C^+}{\Gamma^-; \Omega, P \xrightarrow{L} C^+}\ \mathrm{shift}_P$$

$$\frac{\Gamma^-, A \supset B; \Omega \xrightarrow{L} C^+}{\Gamma^-; \Omega, A \supset B \xrightarrow{L} C^+}\ \mathrm{shift}_\supset$$

4. End:

   (a) We will end-up with a sequent where Ω is empty. Its time to make a choice by applying non-invertible rules.

$$\frac{\Gamma^-;\cdot \xrightarrow{R} A}{\Gamma^-;\cdot \xrightarrow{R} A \vee B}\vee R_1 \qquad \frac{\Gamma^-;\cdot \xrightarrow{R} B}{\Gamma^-;\cdot \xrightarrow{R} A \vee B}\vee R_2$$

$$\frac{\Gamma^-, A \supset B;\cdot \xrightarrow{L} A \quad \Gamma^-; B \xrightarrow{L} C^+}{\Gamma^-, A \supset B;\cdot \xrightarrow{L} C^+}\supset L$$

   (b) After a choice is made, we try to go back to an inversion phase by using the sequent arrow corresponding to where auxiliary formuals went.

   (c) If it is failed at one choice, then we can backtrack to choice points.

Annotation 189 上述算法的核心我们称其为focuing，它是 proof search 中非常重要的一个环节.

3.5 Contraction-free

Annotation 190 在 simplified sequent calculus 中 left implication rule 里面的 left premise 还是留着 main formula, 即

$$\frac{\Gamma, A \supset B \to A \quad \Gamma, B \to C}{\Gamma, A \supset B \to C} \supset L$$

这时你如果考虑对上面的 left primise 再使用一下 $\supset L$，你会发现又得到了 $\Gamma, A \supset B \to A$，这意味你可能陷入在 proof searching 中陷入 loop. 为了尝试解决这个问题，同时让我们的 proof searching 变成更加的 decidable，我需要分解 $\supset L$，以归纳 A 的结构为切入点.

Example 191 如果 A 是 atom，那么此时的 $\subset L$ 对应的 instance 为

$$\frac{\Gamma, a \supset B \to a \quad \Gamma, B \to C}{\Gamma, a \supset B \to C} \supset L$$

这里分三种情况讨论 left premise 的 subproof:

1. 如果 $a \in \Gamma$, 那么显然我们是可以 close 掉 left subproof，直接以 right premise 作为前提.
2. 如果继续对 left premise 继续使用 $\subset L$，前面说了没有意义，更何况这里 A 还只是一个 atom. 那么什么时候重复 apply $\subset L$ 有意义呢? 除非我们还想从 A 里面提取点其他的信息，例如 $A = A_1 \vee A_2$，可以两次 apply $\subset L$，再利用一下 $\vee R_1$ 和 $\vee R_2$ 就能得到不一样的信息.
3. 如果从 Γ 里面挑一个 formula 出来 apply rules，那么这里可能有两种选择，要么 apply invertible rule 或者 non-invertible rule. 对于 non-invertible rule 而言，这里只能选择 $\supset L$，因为此时 right side 是一个 atom. 当我们考虑 delay applying $\supset L$ for $a \supset B$，apply invertible rule and other left implicaion as possible we can, 我们最终会回到第一种情况.

因此我们分析告诉我们在这种情况下，我可以将 $\supset L$ 化简为

$$\frac{\Gamma, a, B \to C}{\Gamma, a, a \supset B \to C}\ a \supset L$$

注意我们在 conclusion 中显式的标注 a 的存在，才能 closed 掉 left branch.

Example 192 如果 A 是 $A_1 \wedge A_2$，考虑以下 logical equivalent:

$$(A_1 \wedge A_2) \supset B \equiv A_1 \supset (A_2 \supset B)$$

为什么要这样换呢？可以对比一下下面的两个 derivations:

$$\dfrac{\dfrac{\Gamma, (A_1 \wedge A_2) \supset B \to A_1 \quad \Gamma, (A_1 \wedge A_2) \supset B \to A_2}{\Gamma, (A_1 \wedge A_2) \supset B \to A_1 \wedge A_2} \wedge R \quad \Gamma, B \to C}{\Gamma, (A_1 \wedge A_2) \supset B \to C} \supset L$$

$$\dfrac{\Gamma, A_1 \supset (A_2 \supset B) \to A_1 \quad \dfrac{\Gamma, A_2 \supset B \to A_2 \quad \Gamma, B \to C}{\Gamma, A_2 \supset B \to C} \supset L}{\Gamma, A_1 \supset (A_2 \supset B) \to C} \supset L$$

很显然后者我们将所有 antecedent 里面 implication 都简化了，每个 implication 都只依赖一个 assumption，这样是有利于我们做归纳的. 因此在这里情况下可以简化 $\supset L$ 为

$$\dfrac{\Gamma, A_1 \supset (A_2 \subset B) \to C}{\Gamma, (A_1 \vee A_2) \subset B \to C} \wedge \subset L$$

Example 193 若 A 为 $A_1 \vee A_2$，这里依然使用一个 logical equivalent:

$$(A_1 \vee A_2) \supset B \equiv (A_1 \supset B) \wedge (A_2 \supset B)$$

这样我们就不需要再使用 $\vee R$ 来从 $A_1 \vee A_2$ 获得信息，因此这里简化的 $\subset L$ 为

$$\dfrac{\Gamma, A_1 \supset B, A_2 \supset B \to C}{\Gamma, (A_1 \vee A_2) \supset B \to C} \vee \supset L$$

Example 194 若 A 为 $A_1 \supset A_2$，我们可以尝试先 derivation 一下

$$\dfrac{\dfrac{\Gamma, (A_1 \supset A_2) \supset B, A_1 \to A_2}{\Gamma, (A_1 \supset A_2) \supset B \to A_1 \supset A_2} \supset R \quad \Gamma, B \to C}{\Gamma, (A_1 \supset A_2) \supset B \to C} \supset L$$

其中这里有一个 logical equivalent:

$$(A_1 \supset A_2) \supset B \wedge A_1 \equiv (A_2 \supset B) \wedge A_1$$

因此这里简化的 $\supset L$ 为

$$\dfrac{\Gamma, (A_2 \supset B), A_1 \to A_2 \quad \Gamma, B \to C}{\Gamma, (A_1 \supset A_2) \supset B \to C} \supset\supset L$$

Example 195 若 A 为 $\top$，显然有

$$\dfrac{\Gamma, B \to C}{\Gamma, \top \subset B \to C} \top \supset L$$

因为 left branch $\Gamma, \top \subset B \to \top$ 显然是可以直接 close 掉的.

Example 196 若 A 为 $\bot$，这里有一个显然的 logical equivalent:

$$\bot \subset B \equiv A$$

因此这里简化的 $\supset L$ 为

$$\frac{\Gamma \to C}{\Gamma, \bot \subset B \to C} \bot \supset L$$

Definition 197 We call simplified sequent calculus with above compound left rules except original left implication G4ip: Contraction-free calculus for intuitionistic logic.

Theorem 198 (Soundness) If the sequent $\Gamma \to C$ is derivable in G4ip, then it is derivable in sequent calculus.

Proof 首先将 G4ip 中的 derivation 写成二维的形式 $\Gamma \overset{\mathcal{D}}{\to} C$，对 $\mathcal{D}$ 做 structure induction. 首先是 bases cases，即 $\mathcal{D}$ 为 empty 的 cases，显然这些 cases 对应的 rules 在 Gi4p 和 sequent calculus 中是保持一致. 再假设 $\mathcal{D}, \mathcal{E}$ is a proof，其对应在 sequent calculus 中的 proof 为 $\mathcal{D}', \mathcal{E}'$. 下面就是证明以它们为 subproof 的 cases，其中只需要证明关于 implication rules，整个过程是非常 straightforward. □

Annotation 199 要证明 completeness of derivability in G4ip，我们得先证明 terminitation of proof searching in G4ip，因为现在没有 soundness 中的前提条件了. 那是什么原因导致我们会担心会不会 terminated 呢? 因为 $\wedge \supset L$

$$\frac{\Gamma, A_1 \supset (A_2 \subset B) \to C}{\Gamma, (A_1 \vee A_2) \subset B \to C} \wedge \subset L$$

看起来 premise 和 conclusion 似乎有相同的规模，这里能否保证 termination? 这里引入 weight 来证明它，很奇妙的规则，至少我还没有洞悉它.

Definition 200 (Weight) For each propositional formula A, we assign it a weight as follow:

- $w(A) = w(\top) = w(\bot) = 2$ for atomic A;
- $w(A \wedge B) = w(A)(1 + w(B))$;
- $w(A \vee B) = 1 + w(A) + w(B)$;
- $w(A \supset B) = 1 + w(A)w(B)$.

Lemma 201 For each rule in G4ip, its premises have a strictly lower weight than its conclusion.

Proof 需要每个例子都验证一遍. 这里我只验证一下上面提到的例子.

$$\begin{aligned} w(A_1 \supset (A_2 \supset B)) &= 1 + w(A_1)(1 + w(A_2)(B)) \\ &= 1 + w(A_1) + w(A_1)w(A_2)w(B) \end{aligned}$$

$$\begin{aligned} w((A_1 \wedge A_2) \supset B) &= 1 + w(A_1)(1 + w(A_2))w(B) \\ &= 1 + w(A_1)w(B) + w(A_1)w(A_2)w(B) \end{aligned}$$

其中 $w(A_1) \geq 2$，因此 $w(A_1 \supset (A_2 \supset B)) \leq w((A_1 \wedge A_2) \supset B)$. □

Theorem 202 (Termination) Proof search in G4ip is terminating

Proof It is straightforward by Lemma 201. □

Theorem 203 (Completeness) If the sequent $\Gamma \to C$ is provable in sequent calculus, then it is provable in G4ip.

Proof 我们需要更细粒度的 structure induction 来处理 $\supset L$，因此我们这里采用induction on the weight of the sequent，这是非常重要的一个 trick. 对于给定一个 sequent，我们得思考 sequent bigger than it，同时我们最好以 proof searching 的视角去看待，这样会让我们讨论的 proof 更加的具体. 这里记录几个 representative cases.

Case 1 若 $\Gamma = \Gamma', A \wedge B$

根据 proof searching 中 invertibility 的策略，可以得到一个 provable premise $\Gamma', A, B \to C$，而它的 weight 小于原 sequent，因此这里可以使用 induction hypothesis 得到它在 G4ip 也是 provable, 接着再使用 $\wedge L$ in G4ip 即可.

Case 2 若 $\Gamma = \Gamma', a, a \supset B$

这里你会发现 sequent calculus 里面没有 rules 可以用，但是我们可以站在更高的一点地方来看它. 如果 $\Gamma', a, a \supset B \to C$ is provable, 那么存在一个 derivation $\mathcal{D}$ end with it，若我们可以从这个 derivation 推出 $a \supset L$ 的 premise，我们就可以正常使用 $a \supset L$ 得到我们想要的结果，这实际就是要说明 $a \supset L$ is invertible.

$$\dfrac{\dfrac{\dfrac{}{\Gamma, a, B, a \to B}\, id}{\Gamma, a, B \to a \supset B}\, \supset R \qquad \begin{matrix}\mathcal{D} \\ \Gamma, a, a \supset B \vdash C\end{matrix}}{\Gamma', a, B \to C}\, cut$$

4 Logical Programming

4.1 Backward Chaining

Annotation 204 Proof searching as computation

Definition 205 We define Horn clauses includes two classes: G denotes goal clauses and P denotes program clauses. They are defined by the following grammer (where A denotes an atom):

$$G \coloneqq A \mid G \wedge G \mid \exists x.G$$
$$P \coloneqq A \mid B \supset A \mid \forall x.P$$
$$B \coloneqq A \mid B \wedge B$$

In words: goals are existentially quantified conjunction of atoms. Programs are either atoms or implications where the antecedent is conjunction of atoms and succedent is an atom, both universally quantified.

Definition 206 A logic program is a sequent $\mathcal{P} \to G$ where the $\mathcal{P}$ contains only program clauses and G is a goal clause.

Example 207 给定下述 logic program:

$$\begin{array}{c} \forall x.plus(0,x,x), \\ \forall x.\forall y.\forall z.plus(x,y,z) \supset plus(s(x),y,s(z)) \end{array} \to \exists x.plus(s(0),s(s(0)),x)$$

其中 $plus$ 是一个 predicate symbol, 而 0 和 s 是 function symbols. 如果我们希望上述 logica program processing(computing) like proof searching in sequent calculus，即以 proof searching 的方式去构造这个 sum. 那么首先我们得思考要构建一个怎样的 proof system?

Definition 208 (Representative proof system) Each clause will be represented as a rule: atoms are rules without premises, implications are rules where the premises are the atoms in B and the conclusion is the succdent. The program computing sums we had before will thus become:

$$\frac{}{plus(0,X,X)}\ plus_0 \qquad \frac{plus(X,Y,Z)}{plus(s(X),Y,s(Z))}\ plus_s$$

Annotation 209 回到 Example 207，我们可以构造一个 proof 说明 x 可以为 $s(s(s(x)))$.

$$\dfrac{\dfrac{}{plus(0,s(s(0)),s(s(0)))}\ plus_0}{plus(s(0),s(s(0)),s(s(s(x))))}\ plus_s$$

Definition 210 Bottom-up proof search is called backward chaining; top-down proof search is called forward chaining.

Annotation 211 Backward chaining 指的就是从 conclusion 推 hypotheses，这个过程包含的几个重要环节为:

1. 从 conclusion 找对应的 inference rules，这个过程我们称为pattern match.
2. 如果某个地方 proof searching 进行不下去了导致 failed，需要回到到某个点重新选择 inference rules，或者没有这样点导致整体完全 failed. 这个过程我们称为backtracking.
3. 以 Exmaple 207为例，它的 conclusion 含有一个未知的 variable x，那这个时候该如何进行第一步呢? 我们首先用一个 symbol X 来占位，那么此时 conclusion 为 $plus(s(0), s(s(0)), X)$，现在它其实可以叫做我们当前的 goal. 此时我们只有第二个 program clause 可以用，因为第一个 program clause require 第一个数得是 0. 那么问题又来了

$$\frac{plus(0, s(s(0)), ?)}{plus(s(0), s(s(0)), X)}\ plus_s$$

   这里我们也可以用另外一个 symbol Y 来占位，即 $plus(0, s(s(0)), Y)$，此时的 goal 只有第一个 program clause 可以对应，同时我必须将 Y 替换为 $s(s(0))$，因为此时没有 far premises 了. 将占位 symbols 替换为指定的 terms 过程我们就称为unification. 当 Y 替换之后，X 配合 $plus_s$ rule 就自然地推出来了. 在最开始 X 这里，我们还不能进行 unify 操作，因为还不是那么明显，因此我们可以 delay it. 后续我们将详细介绍 unification 操作.
4. 如果 apply 某个 inference rule 得到了多个 premises，这样我们可能有多个 goal 需要去 resolve，这里采用depth-first的手法.

Example 212 为什么 disjunction 没有出现在上面的 Horn clauses 里面呢?我们可以来看一个 derivable sequent:

$$p(a) \vee p(b) \to \exists x.p(x)$$

这里我们显然是无法找到 x 满足 $p(x)$，那么我们的 unify 操作在这里是行不通的.

4.2 Prolog

Definition 213 Prolog is a logic programming language implementing backward chaining on Horn clauses.

Example 214 一段 Prolog 程序实际上就是一堆 clauses, 这些 clauses 可以分为两类:

1. A clause may be a fact we know about the world. For example:

   mother(li, maple).
   father(hu, maple).

   Each of those is an atom(predicate) and they are interpreted as a logical formula: e.g., the first fact is $mother(li, maple) \supset \top$, but why?

2. A clause may be a rule in the shape:

   head :- body

   meaning that head is true if body is true. The head is atom, but the body can be conjunction of atoms. We can have following ruls for family relations, for example:

   parent(X,Y) :- mother(X, Y).
   parent(X,Y) :- father(X, Y).
   sibling(X,Y) :- parent(Z,X), parent(Z,X).

   It also has a representation as logical formulas, for example, the last clause is
   $\forall x.\forall y.\forall z.(parent(z, x) \wedge parent(z, y) \supset sibling(x, y))$.

当我们有了上面这些 clauses 之后, 我们就可以来做一些 querys, 例"maggie 的 parent 是谁?", parent(X, maggie). Prolog 以它作为 current goal, 开始搜索 head 是 parent 的那些 clauses, 找到对应的 body 里面的 subgoals, 接着周而复始, 直到最后 facts, 这过程充斥着我们提到的 backtracking 和 unification.

Example 215 一个非常神奇的例子: quicksort.

```
quicksort([], []).
quicksort([X|Xs], Ys) :- partition(Xs, X, Ls, Gs),
                         quicksort(Ls, Sl),
                         quicksort(Gs, Sr),
                         append(Sl, [X|Sr], Ys).

partition([], _, [], []).
partition([X|L], P, [X|Ls], Gs) :- X < P,
        partition(L, P, Ls, Gs).
partition([X|L], P, Ls, [X|Gs]) :- X >= P,
        partition(L, P, Ls, Gs).
```

配合 trace 理解它:

```
 1    1  Call: quicksort([2,3,1],_29) ?
 2    2  Call: partition([3,1],2,_100,_101) ?
 3    3  Call: 3<2 ?
 3    3  Fail: 3<2 ?
 3    3  Call: 3>=2 ?
 3    3  Exit: 3>=2 ?
 4    3  Call: partition([1],2,_153,_87) ?
 5    4  Call: 1<2 ?
 5    4  Exit: 1<2 ?
 6    4  Call: partition([],2,_140,_87) ?
 6    4  Exit: partition([],2,[],[]) ?
 4    3  Exit: partition([1],2,[1],[]) ?
 2    2  Exit: partition([3,1],2,[1],[3]) ?
 7    2  Call: quicksort([1],_233) ?
 8    3  Call: partition([],1,_259,_260) ?
 8    3  Exit: partition([],1,[],[]) ?
 9    3  Call: quicksort([],_284) ?
 9    3  Exit: quicksort([],[]) ?
10    3  Call: quicksort([],_309) ?
10    3  Exit: quicksort([],[]) ?
11    3  Call: append([],[1],_337) ?
11    3  Exit: append([],[1],[1]) ?
 7    2  Exit: quicksort([1],[1]) ?
```

```
12    2  Call: quicksort([3],_363) ?
13    3  Call: partition([],3,_389,_390) ?
13    3  Exit: partition([],3,[],[]) ?
14    3  Call: quicksort([],_414) ?
14    3  Exit: quicksort([],[]) ?
15    3  Call: quicksort([],_439) ?
15    3  Exit: quicksort([],[]) ?
16    3  Call: append([],[3],_467) ?
16    3  Exit: append([],[3],[3]) ?
12    2  Exit: quicksort([3],[3]) ?
17    2  Call: append([1],[2,3],_29) ?
17    2  Exit: append([1],[2,3],[1,2,3]) ?
 1    1  Exit: quicksort([2,3,1],[1,2,3]) ?
```

其中第一列数字表示具体哪个 call, 第二列数字表示 the depth of goal, 其中带下划线的数字表示占位的 variable, 等待被 unified. 可以看到第 8 次 call 的时候第一次 unify.

4.3 Focusing

Annotation 216 首先扩充一下 3.4小节提到的 invertible calculus 到 first order logic. 这里我们只记录新引入的 inference:

Right invertible rules	$\dfrac{\Gamma^-;\Omega \xrightarrow{R} A[c/x]}{\Gamma^-;\Omega \xrightarrow{R} \forall x.A}\ \forall R$ where c is fresh variable
Side change	$\dfrac{\Gamma^-;\Omega \xrightarrow{L} \exists x.A}{\Gamma^-;\Omega \xrightarrow{R} \exists x.A}\ LR_\exists$
Left invertible rules	$\dfrac{\Gamma^-,\Omega,A[c/x] \xrightarrow{L} C^+}{\Gamma^-,\Omega,\exists x.A \xrightarrow{L} C^+}\ \exists L$ where c is fresh variable
Shift rules	$\dfrac{\Gamma^-,\forall x.A;\Omega \xrightarrow{L} C^+}{\Gamma^-;\Omega,\forall x.A \xrightarrow{L} C^+}\ shift_\forall$
Noninvertible rules	$\dfrac{\Gamma^-;\cdot \xrightarrow{L} A[t/x]}{\Gamma^-;\cdot \xrightarrow{L} \exists x.A}\ \exists R$ $\quad$ $\dfrac{\Gamma,\forall x.A;A[t/x] \to C^+}{\Gamma,\forall x.A;\cdot \to C^+}\ \forall L$ where t is term

这里有两个感觉很奇怪的 rules $L\exists$ 和 $L\forall$，首先这和我们第一直觉上关于 $\exists$ 和 $\forall$ 的 introduction rule 不太一样，而 $R\exists$ 和 $R\forall$ 是符合我们直觉的. 我们从“怎么用这两个条件”来解释:

- 在一般情况下，如果我们有一个关于 type τ 存在性命题，那么通常我们会假设其存在用某个 symbol 满足 τ 同时满足当前命题，让 proving 过程进行下去，$L\exists$ 就是采用的这种直觉.
- 而当存在一个关于 type τ 全称命题的时候，那么我们会取一个任意满足 τ 的 term，来让 prove 过程进行下，同时 keep 这个全称命题，万一取错了，我们还可以再取.

上面是我个人觉得比较合理的解释. 但是当你觉得上述解释 make sense 之后，你回过来看 $\exists R$ 和 $\forall R$ 好像又有点不对劲，终于你会了解核心所在是因为 $\exists$ 和 $\forall$ 出现的位置不一样，出现在 primise 就是上面解释. 而当出现在 conclusion 之后，$\exists x$ 意味着我们需要一开始就找到符合命题的 term, 最终才能得到正确的 proof; $\forall x$ 意味我们 x 怎样取都可以，因此我们可以用一个 free variable 来代替.

Definition 217 Positive connectives are those that have invertible left rules and non-invertible right rules.

Definition 218 Negative connectives are those that have non-invertible left rules and invertible right rules.

Annotation 219 那么 $\vee, \bot, \exists$ 都是 positive，而 $\supset, \forall$ 都是 negative. 至于 $\wedge$ 和 $\top$ 它们是 both positive and negative，我们称其为neutral.

Annotation 220 利用 polarity 的 defintion，我们可以做一些 generalization. 将 $LR_P, LR_\vee, LR_\perp, LR_\exists$ 可以统一为

$$\frac{\Gamma^-; \Omega \xrightarrow{L} C_a^+}{\Gamma^-; \Omega \xrightarrow{R} C_a^+}\ LR$$

类似地，可以将 $shitf_P, shitf_\supset, shitf_\forall$ 统一为

$$\frac{\Gamma^-, C_a^-; \Omega \xrightarrow{L} C_a^+}{\Gamma^-; \Omega, C_a^- \xrightarrow{L} C_a^+}\ shift$$

Definition 221 we define *focus* rule for deciding which formula to focus on , after exhausting all invertible rule on the rule and on the right.

$$\frac{\Gamma^-; \cdot \xrightarrow{R} [P]}{\Gamma^-; \cdot \xrightarrow{L} P}\ focus_r \qquad \frac{\Gamma^-, N; [N] \xrightarrow{R} C_a^+}{\Gamma^-, N; \cdot \xrightarrow{L} C_a^+}\ focus_r$$

where $[F]$ donates which non-invertible formula in focus, P is positive formula and N is negative formula.

Definition 222 We define *synchronous* rule as follow:

$$\frac{\Gamma^-; \cdot \xrightarrow{R} [A]}{\Gamma^-; \cdot \xrightarrow{R} [A \vee B]}\ \exists R_1 \qquad \frac{\Gamma^-; \cdot \xrightarrow{R} [A]}{-\ \Gamma^-; \cdot \xrightarrow{R} [A \vee B]}\ \exists R_2 \qquad \frac{\Gamma^-; \cdot \xrightarrow{R} [A[t/x]]}{\Gamma^-; \cdot \xrightarrow{R} [\exists x.A]}\ \exists R_1$$

$$\frac{\Gamma^-; \cdot \xrightarrow{R} [A] \quad \Gamma^-; [B] \xrightarrow{L} C_a^+}{\Gamma^-; [A \supset B] \xrightarrow{L} C_a^+}\ \supset L \qquad \frac{\Gamma^-; [A[t/x]] \xrightarrow{L} C_a^+}{\Gamma^-; [\forall x.A] \xrightarrow{L} C_a^+}\ \forall L$$

Definition 223 We define *blur* rules to remove focus as follow:

$$\frac{\Gamma^-; \cdot \xrightarrow{R} N}{\Gamma^-; \cdot \xrightarrow{R} [N]}\ blur_r \qquad \frac{\Gamma^-; P \xrightarrow{L} C_a^+}{\Gamma^-; [P] \xrightarrow{L} C_a^+}\ blur_r$$

where N is negative formula and P is positive formula.

Definition 224 We define new *init* rules under focus as follow:

$$\frac{}{\Gamma^-; [a^-] \xrightarrow{L} a^-}\ init_L \qquad \frac{a^+ \in \Gamma^-}{\Gamma^-; \cdot \xrightarrow{R} [a^+]}\ init_R$$

Annotation 225 注意在224中给 atoms 也添加上了 polarity，本质上 atoms 可以是 positive，也可以是 negative. 在 *init* rules 为了确保 focus $[F]$ 保持原本的含义，因此对应的 atom 的 polarity 也要保持一致.

Definition 226 The system with all above rules called first logic focused calculus.

Annotation 227 引入 *focus* rules 会造成整体的 the shape proof searching 发生变化，当你 made a choice 之后选择了某个 non-invertible formula，focus 操作会强制使得这个 non-invertible formula 直到被分解成了 invertible subformula 才会去掉 focus. 同时 *init* rule 只有在 atom 在 focused 的时候才有可能被 applied. 对比3.4节的 proof searching algorithm，我们可以在 make a choice 之后，如果遇到无法处理的 subformula is atom on left, 我们可以做一下 *shift* 操作把它放到 Γ^- 里面去，转而处理其他的 formula. 但是在这里不行你如果碰到了无法处理的 focused atom，你就得 backtracking 了，这就是最大的区别. 然而 backtracking 也是技术的，我们需要恢复到上述最早 focused non-inverible formula，然后换个 non-invertible 来继续 focus.

Annotation 228 这里我们想来解释一下 definition 222里面的synchronous的含义. 字面意思同步处理，从另外一个方式思考那么对 invertible formula 的处理就叫 asynchronous rules，即异步处理. 试想如果现在拿掉 side change rules，两个都各自有一个 invertible formula，实际上谁先谁后实际并没有太本质的区别，这就是 asynchronous 的含义. 而对 non-invertible formula 的处理，我们需要来一个一个的做选择，一个不行就换下一个，这就是 synchronous 的含义.

Definition 229 (Simpifiled focus calculus) The system gather all inveritible rules to asynchronous rules and removes side change rule based on first logic focus calculus is called LJF

4.4 Polarities

Annotation 230 这小节将通过一个例子解释 how uniform proof works 和为什么赋予 atoms 不同的 polarity 将导致 shape of proof searching.

Example 231 给定一个 non-standard(inlined arithmetic op) prolog program of Fibonacci:

```
fib(0, 0).
fib(1, 1).
fib(n+2, f2 + f2) :- fib(n + 1, f1), fib(n, f2).
```

然后我们需要想要证明 `fib(5, 5)`，在逻辑上等价于证明下述 sequent

$$\begin{array}{c} fib(0,0), fib(1,1), \\ \forall n.\forall f_1.\forall f_2.(fib(n+1,f_1) \wedge fib(n,f_2) \supset fib(n+2, f_1+f_2)) \end{array} \longrightarrow fib(5,5)$$

这里我们将其放到 LJF229中来证明.

- 首先以

  $$\cdot; \Gamma \to fib(5,5)$$

  作为 end-sequent. 这里只能首先 apply *shift* rule 把 Γ 放到 Γ^-，因为两边都没有办法使用 synchronous and asynchronous rules. 这里需要连续使用 3 次 *shift*，得到

  $$\begin{array}{c} \Gamma; \cdot \to fib(5,5) \\ shift \times 3 \\ \cdot; \Gamma \to fib(5,5) \end{array}$$

- 此时我们如果对 atoms assign different polarity 将会使得从不同的方向继续进行证明:

  1. 假设atoms are negative，那么显然 right atom $fib(5,5)$ 是无法被 focused，只能依次 focus Γ^- 里面的 formulas. 对于 $fib(0,1)$ 和 $fib(1,1)$，这里 $init_l$ 显然是用不了，因为它们并没有在 right side 出现. 那么只能选择最后的 universal quantified formula 了.
  2. 假设atoms are positive formula，那么这里 right atom 就可以被 focus 了，但是这里对应的 $init_r$ 还是用不了. 因此最后也能选择 left side's universal quantified formula.

- 此时我们 focus left side's universal quantified formula 并且 apply 3 times $\forall L$，则有

  $$\dfrac{\begin{array}{c} \Gamma; [(fib(n+1,f_1), fib(n,f_2) \supset fib(n+2,f_1+f_2))] \to fib(5,5) \\ \forall L \times 3 \\ \Gamma; [\forall n.\forall f_1.\forall f_2.(fib(n+1,f_1) \wedge fib(n,f_2) \supset fib(n+2,f_1+f_2))] \to fib(5,5) \end{array}}{\begin{array}{c} \Gamma; \cdot \to fib(5,5) \\ shift \times 3 \\ \cdot; \Gamma \to fib(5,5) \end{array}} focus_l$$

- 此时依然处于 focus 当中，继续 apply $\supset L$，则有

$$\dfrac{\dfrac{\Gamma;\cdot \to [fib(n+1,f_1)\wedge fib(n,f_2)] \quad \Gamma;[fib(n+2,f_1+f_2)]\to fib(5,5)}{\Gamma;[(fib(n+1,f_1)\wedge fib(n,f_2)\supset fib(n+2,f_1+f_2))]\to fib(5,5)}\supset L}{\dfrac{\Gamma;[\forall n.\forall f_1.\forall f_2.(fib(n+1,f_1)\wedge fib(n,f_2)\supset fib(n+2,f_1+f_2))]\to fib(5,5)}{\Gamma;\cdot\to fib(5,5)}\begin{array}{c}\forall L\times 3\\ focus_l\end{array}}$$
$$shift\times 3$$
$$\cdot;\Gamma\to fib(5,5)$$

- 其中 left branch 现在可以使用 $blur_r$ 去掉 focus 了，则有

$$\dfrac{\dfrac{\dfrac{\dfrac{\Gamma;\cdot\to fib(n+1,f_1) \quad \Gamma;\cdot\to fib(n,f_2)}{\Gamma;\cdot\to fib(n+1,f_1)\wedge fib(n,f_2)}\wedge R}{\Gamma;\cdot\to [fib(n+1,f_1)\wedge fib(n,f_2)]}blur_r \quad \Gamma;[fib(n+2,f_1+f_2)]\to fib(5,5)}{\Gamma;[(fib(n+1,f_1)\wedge fib(n,f_2)\supset fib(n+2,f_1+f_2))]\to fib(5,5)}\supset L}{\dfrac{\Gamma;[\forall n.\forall f_1.\forall f_2.(fib(n+1,f_1)\wedge fib(n,f_2)\supset fib(n+2,f_1+f_2))]\to fib(5,5)}{\Gamma;\cdot\to fib(5,5)}\begin{array}{c}\forall L\times 3\\ focus_l\end{array}}$$
$$shift\times 3$$
$$\cdot;\Gamma\to fib(5,5)$$

- 从这里开始就可以变得有趣了，注意三个标蓝色的formulas现在都是 atoms 同时它们都是被 variables 包裹了，我们就可以对其做 unify 了，但是不同 polarity of atoms 将构造完全的不同的 proof searching.

  1. 如果atoms are negative，那么最终只能用 $init_l$ 来 close 掉 proof. 你仔细观察最左边的 branch，这里是可以用 $init_l$ 将其 close 掉，因为其中 variables assign 都是 unified. 这里必有 $n=3$，可 f_1 和 f_2 还是确定不了，但是依然可以减少它们 searching space，即 $f_1=x, f_2=5-x$，那么最右边 branch 也因此被 close 了. 此时最上面两个 branches 就变成了

  $$\Gamma;\cdot\to fib(4,x) \text{ and } \Gamma;\cdot\to fib(3,5-x) \qquad \text{iter-con}$$

  它们和我们 end-sequent 是非常像的，我们可以用前面的步骤分别在对它们再做一次 derivation，对于 $fib(4,x)$，利用 iter-con 显然我们可以得到

  $$\Gamma;\cdot\to fib(3,y) \text{ and } \Gamma;\cdot\to fib(2,x-y)$$

  对于 $fib(3,5-x)$ 同理有

  $$\Gamma;\cdot\to fib(2,z) \text{ and } \Gamma;\cdot\to fib(1,5-x-z)$$

  可以看到直到 $fib(1,0)$ 或者 $fib(0,0)$ 才会停止. 这背后本质就是如果我要计算第 5 个 Fabonacci number, 那么我得先计算第 4 个 Fabonacci number 和第 3 个 Fabonacci number.

2. 如果atoms are positive，那么最终只能用 $init_r$ 来 close 掉 proof. 那么此时我们是可以 focus 最上面两个 branch 的 right side atom，这两个 atom 里面的 variables 也都是 unified，同时我们可以再利用 $init_r$ 来 close 掉这两个 branch，此时得到 $n=0, f_1=1, f_2=0$. 此时只剩下了最右边的 branch，即

$$\Gamma; fib(2,1) \to fib(5,5)$$

相比于 end-sequent，我们多了一个 knowledge $fib(2,1)$. 按照这种思路我们可以继续得到 $fib(3,2)$，依次往后类推.

Annotation 232 想象一下我们正常计算第 $n+2$ 个 Fabonacci number 的思路，无非两种:(1) 先计算第 $n+1$ 和第 n 个，它们之和就是第 $n+2$ 个; (2) 从第 0 个开始计算起，依次往后计算 $1,2,3,\cdots$ 直到 $n+2$ 个. 它们正好就对应了 atoms are negative 和 atoms are positive 的情况. Prolog 实际上就应用了第一种思路，这种思路叫goal-oriented strategy.

在 Prolog world 里面我们的 query with head of clause and continue with the body as the new goal，在 proof world 里面它就对应了 close the right branch after $\supset L$ and continuing with the left. 这好像看起来在上面的例子中没那么显然，那是因为用了不正规的 Prolog，即 inlined arithmetical express，正常来说我们得这样写

$$fib(M,N) :- \ fib(X_1,Y_1), X_2 = X_1+1, fib(X_2,Y_2), M = X_1+2, N = Y_1+Y_2$$

与思路 2 对应的就是前面我们提到的 forward chaining，在这种情况 left branch of $\supset L$ must be closed 来产生新的 knowledge.

Annotation 233 前面例子里面有一个小细节就是，里面用的 variables 都是暗含了 universal quantifier. 这一点需要注意，同时 unification 是具有传递过程的，比如在某个地方确定了某个 variable 的值，那么在它前面的，也就是从 shape of proof 向下含有 x 的地方，也要具有相同的值.

4.5 Foward Chaining

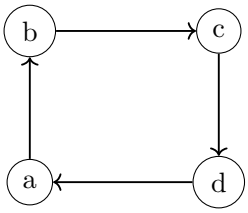
Definition 234 The state that no new facts can be generated from the sef of known facts is called saturation.

Example 235 假设我们下面的一个小 program snippet:

```
path(X,Y) :- edge(X,Y).
path(X,Z) :- edge(X,Z), path(Z,Y).

cycle :- path(X,X).
```

同时给定一个 graph:




```
edge(a,b).
edge(b,c).
edge(c,d).
edge(d,a).
```

那么一个正常的 forward chaining engine 就会根据上面的 logic program 产生新关于 path 的 knowledges，后续以此产生 cycle.

Definition 236 The set of known facts is called database.

Annotation 237 有两个比较在 forward chaining 里面比较重要的概念:

- Facts are always ground atoms, meaning that there no free variable lingering around. 因为在 forward logic programming 里面 head 里面出现的变量一定都是在 body 里面的，因此这样你新产生的 fact 也是不带 free variables 的.
- The database is consiedered as a set, so there is no duplication of facts in it.

Example 238 接下来我们看一个有趣的应用，我们可以把 forward chaining 应用到 programming language 的 parser 上. 以 context-free grammar 为例，一个 context-free grammar 可以表示为一个 tuple $G = (V, \Sigma, \mathcal{P}, S)$，其中 V 表示各种 non-terminal symbols 集合，Σ 表示各种 terminators，$\mathcal{P}$ 表示各种 production rules of the form $X \Rightarrow \gamma (X \in V$ and γ is a string with other non-terminal symbols or terminator)，最后 S 表示 start symbol.

对于一个只包含 terminator 的 string w，它在 G 中一个 derivation 可以表示为 $S \Rightarrow \gamma_1 \cdots \Rightarrow \gamma_n \Rightarrow w$，其中每一步可以看做从 γ_i 中挑一个 non-terminal symbol X 出来，和 X 对应的一个 production rule $X \Rightarrow \alpha$，将 γ_i 里面所有 X 出现的地方都换成 α，那么就得到了 γ_{i+1}. 有一个很自然的问题就是如何把这样的 derivation 写成 proof 的形式呢?

首先我们得构造类似的 inference rules，我们用 $\gamma \Rightarrow^* w$ 表示 w 可以由 γ 推导得来. 这里给分别给出三种 inference rules:

$$\frac{\gamma_1 \Rightarrow^* w_1 \quad \gamma_2 \Rightarrow^* w_2}{\gamma_1\gamma_2 \Rightarrow^* w_1w_2}\ conc \qquad \frac{}{a \Rightarrow^* a}\ id \qquad \frac{\gamma \Rightarrow^* w}{X \Rightarrow^* w}\ (r)X \Rightarrow \gamma$$

其中 a 是 terminator. Rule *conc* 提供了 split string 的方法，rule *id* 适配 terminator，最后 rule (r) 实际上一组 rules，对应每一个 production rules 生成.

例如我们给定一个 context-free grammar:

$$\begin{aligned} T &\Rightarrow -T && [neg] \\ &\mid T+T && [plus] \\ &\mid (T) && [pars] \\ &\mid var && [var] \\ &\mid 1 && [one] \\ &\mid 0 && [zero] \end{aligned}$$

我们来看一下关于 $-(x+1)$ 的 derivation，即我们要找到关于 $T \Rightarrow^* -(x+1)$ 的 proof，这里的 S 就是 T.

$$\dfrac{\dfrac{}{- \Rightarrow^* -}\ id \qquad \dfrac{\dfrac{\dfrac{}{(\Rightarrow^* (}\ id \qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{x \Rightarrow^* x}\ id}{T \Rightarrow^* x}\ var \qquad \dfrac{\dfrac{}{+ \Rightarrow +}\ id \qquad \dfrac{1 \Rightarrow^* 1}{T \Rightarrow^* 1}\ one}{+T \Rightarrow^* +1}\ conc}{T+T \Rightarrow^* x+1}\ conc}{T \Rightarrow^* x+1}\ plus \qquad \dfrac{}{) \Rightarrow^*)}\ id}{T) \Rightarrow^* x+1)}\ conc}{(T) \Rightarrow^* (x+1)}\ conc}{T \Rightarrow^* (x+1)}\ pairs}{-T \Rightarrow^* -(x+1)}\ conc}{T \Rightarrow^* -(x+1)}\ neg$$

你应该可以隐约的感受到如何选择 substring 应用 *conc* 是关键中的关键的.

接下来我们可以想想如何应用 forward chaining 呢？首先我们要给出一些 basic facts，那么它们应该是什么呢？同时我们应当如何利用这些 basic facts 生成更多我们感兴趣的 facts 呢？你要知道利用上述 gammar 是可以生成 infinite string matched that gammar. 如果我们想要 query $T \Rightarrow^* w_0$，这里的做法是

1. 把 w_0 中的 terminator 作为 basic facts($a \Rightarrow^* a$ for every terminator a in w_0)
2. 我们只 apply 满足下述条件 $\gamma \Rightarrow w$ (γ is head and w is body):
   (a) γ is the substring of the right-side of gammar rules(i.e., terminator, T, $T+T$, $-T$ and (T))
   (b) w is the substring of w_0

第一个条件是为了避免我们生成语法之外的 gammer snippets; 第二个条件显然是为了 goal-orient. 你可能会想只要第二个不行吗? 肯定是不行的，因为从 w_0 里面取的 substring 可能并不符合 gammar，例如你从 $-(x+1)$ 取 $-(\Rightarrow^* -($，你对它只能继续 apply $conc$，但是这没有任何意义，你后续可能无法 apply (r) rules 把 left side 变成 T，我们的目标是 $T \Rightarrow^* w_0$. 使用这种方法对 $-(x+1)$ 做 forward chaining 为

1	$-$	$\Rightarrow^*$	$-$	
2	$($	$\Rightarrow^*$	$($	
3	x	$\Rightarrow^*$	x	
4	$+$	$\Rightarrow^*$	$+$	
5	1	$\Rightarrow^*$	1	
6	$)$	$\Rightarrow^*$	$)$	
7	T	$\Rightarrow^*$	1	$one(5)$
8	T	$\Rightarrow^*$	x	$var(3)$
9	$T+T$	$\Rightarrow^*$	$x+1$	$conc(7,8)$
10	T	$\Rightarrow^*$	$x+1$	$plus(9)$
11	(T)	$\Rightarrow^*$	$(x+1)$	$conc \times 2(2,10,6)$
12	T	$\Rightarrow^*$	$(x+1)$	$pars(11)$
13	$-T$	$\Rightarrow^*$	$-(x+1)$	$conc(1,12)$
14	T	$\Rightarrow^*$	$-(x+1)$	$neg(13)$

你发现这种手法和我们常见的 programming language 前端的 lexer 和 parser 非常的像，lexer 拿到 token 交给 parser，parser 尝试生成的 abstract syntax，amazing!

最后想一个问题为什么在最前面我们已经推出了一个 backward chaining 的方法，还会想把 forward chaining 应用到 parser 上呢? 这是一个好问题，当然是复杂度啊！forward chaining method 会更加的 cheap. 我们来分别计算一下不同方法下的复杂度，首先给定一个一般地 $S \Rightarrow^* w$，其中 $|w| = n$.

其中 backward chaining 的复杂度，你观察它的 derivation 实际上近似一个 full binary tree，其叶子树就是 terminators 的个数 n，这样一棵 tree 那么就有 $2^n - 1$ 个结点. 同时这样 n 个叶子结点的 full binary tree 可能会有很多种，这是因为 apply $conc$ 产生的结果，这样 full binary tree 一共有 $\frac{2n!}{(n+1)!n!}$ 种. 这样我们可以考虑 worst-case，即尝试了所有的 split 思路最后还是失败了，那么这样就产生了所有 full binary tree 都会涉及到.

最后来看看 forward chaining 的复杂度，它的复杂度取决于 saturated database 中的 facts 的个数，以及在 searching 过程中 apply rules 的个数. 首先来看 saturated database 中的 facts 的个数: 每个 fact 形如 $\gamma \Rightarrow^* w'$ 满足 γ is substring of the right side of grammer rules and w' is the substring of w，设当前 context-free gammar 里面 longest right side of all grammar rules 的长度为 k. 同时 w 有 $\frac{(1+n)(n)}{2}$ 个 substring，简单想一下这是

显然的，同理某个 right side of gammar rule 也最多有 $\frac{(1+k)(k)}{2}$ 个 substring，这里感觉还要考虑一下有多少个 production rule，我们设为 m. 那么最多就有 $m \cdot \frac{(1+n)(n)}{2} \cdot \frac{(1+k)(k)}{2} \in O(mk^2n^2)$ 个 facts. 任意地一个 facts，我们可以对其 apply *conc* 或者 (r). 对于 (r) rules 它只有一个 premise，因此我们最多 apply $O(mk^2n^2)$ 次，而对于 *conc* rule 有两个 premise，但是这里是有限制是，因为 $\gamma_1\gamma_2$ 必须还是一个 substring of the right side of gammar rules，同时 w_1w_2 也是 substring of w, 因此当 $\gamma_1 \Rightarrow^* w_1$ 确定之后，$\gamma_2 \Rightarrow^* w_2$ 最多只有 $O(kn)$ 个选择，那么最多可以 apply $O(mk^3n^3)$ 次. 而 m 和 k 都是 finite 的，因此最终为 $O(n^3)$.

当然还有办法优化，如果 k 比较大，可以将我们 context-free gammar 转化成 Chomsky normal form:

$$X \Rightarrow YZ \quad \text{or} \quad X \Rightarrow a.$$

这样 k 就被严格限制为 2.

4.6 Uniform Proofs

Definition 239 A uniform proof is a proof in which each occurrence of a sequent $\Gamma \to C$, where C is a composed formula, is the conclusion of a rule introduing $C's$ *topmost* connective.

Annotation 240 一个 uniform proof 就是尽可能先地 decompose right formula, 即 apply right rule，或者换句话来说它只对 right invertible rules 感兴趣 i.e. the negative and neutral ones.

Definition 241 An *abstract logic programming language* is a logic in which every provable sequent $\Gamma \to C$ has a uniform proof.

Annotation 242 这里可以解释为什么没有类似 $\vee$ 的 connectives 出现在 logic programming 里面，因为 $\vee$ 是 positive 的. 这里还缺点解释 todo.

Annotation 243 关于 unification algorithm 实际上我们已经在 type checking 什么的时候见识过了，这里概念上没什么区别. 但是这里我们会将其变成 inference rule 的形式，最终构建一个完整的 deductive system.

Definition 244 we use the notation $t \doteq s$ to denote the judgment t and s are unifiable.

Definition 245 Given two substitutions σ_1 and σ_2, we define their composition as follow:

$$\sigma_1\sigma_2 = \sigma_2 \circ \sigma_1$$

Lemma 246 (uniqueness of most general unifier) If σ_1 and σ_2 are both the most general unifier, then σ_2 is essentially a variable renaming of σ_1.

Proof 这里我们将 most general unifier 简写成 mgu, 根据 mgu 的 definition，我们知道分别存在 δ_1 和 δ_2 使得

$$\sigma_1 = \sigma_2\delta_1$$
$$\sigma_2 = \sigma_1\delta_2$$

于是 $\sigma_1 = (\sigma_1\delta_2)\delta_1$，这意味着 δ_2 改变的东西，都被 δ_1 还原了. 并且 δ_2 就只能改变做一些 variables maps to variables 的转换，是不能将 variables map to constants，因为 δ_2 是没法还原 constants map to varaibles. 同理 $\sigma_2 = (\sigma_2\delta_2)\delta_1$，其实 δ_1 和 δ_2 互为 inverse. □

Definition 247 we define jugement t and s are unifiable under most general unifier σ as

$$t \doteq s \mid \sigma$$

Definition 248 We define unification calculus as follow

$$\frac{}{x \doteq x \mid \{\}} \qquad \frac{x \notin FV(t)}{x \doteq t \mid \{x \to t\}} \qquad \frac{t = f(\bar{t}) \quad x \notin FV(t)}{t \doteq x \mid \{x \to t\}}$$

$$\frac{\bar{t} \doteq \bar{s} \mid \sigma}{f(\bar{t}) \doteq f(\bar{s}) \mid \sigma} \qquad \frac{}{(\cdot) \doteq (\cdot) \mid \{\}} \qquad \frac{t \doteq s \mid \sigma_1 \quad \bar{t}\sigma_1 \doteq \bar{s}\sigma_1 \mid \sigma_2}{(t, \bar{t}) \doteq (s, \bar{s}) \mid \sigma_1\sigma_2}$$

where x is a variable, t is a term, $\bar{t}$ and $\bar{s}$ are sequence of terms, $(\cdot)$ is empty sequence of terms, $\ $ is empty substitution, f is function, and $\bar{t}\sigma_1 = \sigma_1(\bar{t})$.

Theorem 249 (soundness) If $t \doteq s \mid \sigma$ then $t\sigma = s\sigma$.

Theorem 250 (completeness) If $t\sigma' = s\sigma'$, then $t \doteq s \mid \sigma$ and $\sigma' = \sigma\delta$.

Proof completeness 的证明咋一想还是不是那么显然，这里记录一下. 还是使用 induction hypothesis，但是这里要对 $t\sigma'$ 做 induction，而不是 t，后面我们会看到为什么.

- Base cases:
  - $t\sigma'$ is a variable.

    那么这种情况下 t 也只能是 variable x, 并且 σ' 只能是 $\{x \to z\}$, 其中 z 也是 variable. 考虑 assumption $t \doteq s$，设 $s = y$，那么这里的 $\sigma' = \{x \to z, y \to z\}$. 根据 unification calculus，若 $x = y$ 则 $\sigma = \{\}$, 那么 $\sigma' = \sigma\{x \to z\}$; 若 $x \neq y$，则 $\sigma = \{x \to y\}$，那么 $\sigma' = \sigma\{y \to z\}$.
  - $t\sigma'$ is a empty sequence.

    那么这种情况下 t 也只能是一个 emptys equence，此时 σ' 为任意的 substitution，同理 s 也只能是一个 empty sequence. 此时 $\sigma = \{\}$，因此 $\sigma' = \sigma\sigma'$.
- Induction hypotheses: 这里有两个 induction hypotheses:
  1. If $t\sigma' = s\sigma'$, then $t \doteq s \mid \sigma$ and $\sigma' = \sigma\delta$.
  2. If $\bar{t}\sigma' = \bar{s}\sigma'$, then $t \doteq s \mid \sigma$ and $\sigma' = \sigma\delta$.
- Inductive steps:
  - $t\sigma'$ is a term r that is not a variable. 这里我们分析 t 可能的结构

    Case 1 $t = x$

    Subcase 1 $s = y$

$x\sigma' = y\sigma'$	assumption
$\sigma' = \{x \to r, y \to r\}$	by defintion of substitution
$t \doteq s \mid \{x \to y\}$	by unification calculus(这里实际还需要 cover 一下 $x = y$ 的情况)
$\sigma' = \{x \to y\}\{y \to r\}$	substitution composition

Subcase 2 $s = f(\overline{s})$

$x\sigma' = f(\overline{s})\sigma'$	assumption
$\sigma' = \{x \to f(\overline{s})\sigma'\} \cup \sigma'_1$	def of sub and σ'_1 is σ/x
$x \notin FV(f(\overline{s}))$	otherwise unifiable
$\sigma' = \{x \to f(\overline{s})\sigma'_1\} \cup \sigma'_1$	$f(\overline{s})\sigma' = f(\overline{s})\sigma'_1$
$\sigma' = \{x \to f(\overline{s})\}\sigma'_1$	def of sub comp

Case 2 $t = f(\overline{t})$

Subcase 1 $s = y$ same as *subcase* 1.2.

Subcase 2 $s = f(\overline{s})$

$f(\overline{t})\sigma' = f(\overline{s})\sigma'$	assumption
$\overline{t}\sigma' = \overline{s}\sigma'$	def of sub
$\sigma' = \sigma\delta$	$\overline{t}\sigma'$ 在规模上严格小于 $f(\overline{t})\sigma'$, 因此这里可以用 IH 2
$\overline{t} \doteq \overline{s} \mid \sigma$	by unification calculus
$f(\overline{t}) \doteq f(\overline{s}) \mid \sigma$	by unification calculus
$\sigma' = \sigma\delta$	final

– $t\sigma'$ is non-empty sequence of terms.

Case 1 $t = (t_1, \overline{t}_2)$

Subcase 1 $s = (s_1, \overline{s}_2)$

$(t_1, \bar{t}_2)\sigma' = (s_1, \overline{s}_2)\sigma'$	assumption
$(t_1\sigma', \bar{t}_2\sigma') = (s_1\sigma', \overline{s}_2\sigma')$	def of sub
$t_1 \doteq s_1 \mid \sigma$ and $\sigma' = \sigma\delta_1$	$\lvert t_1\sigma'\rvert < \lvert t\sigma'\rvert$ IH 1
$\bar{t}_2(\sigma\delta_1) = \overline{s}_2(\sigma\delta_1)$	def of sub
$(\bar{t}_2\sigma)\delta_1 = (\overline{s}_2\sigma)\delta_1$	comp of sub
$\bar{t}_2\sigma \doteq \overline{s}_2\sigma \mid \sigma_2$ and $\delta_1 = \sigma_2\delta_2$	$\lvert(\bar{t}_2\sigma)\delta_1\rvert < \lvert t\sigma'\rvert$ IH 2
$\sigma' = \sigma\sigma_2\delta_2$	def of sub

如果这里我们不是对 $t\sigma'$ 做 induction，而是对 t 做 induction，那么这里的 induction hypothesis 就变成了 $|\bar{t}_2\sigma|$，但是这里我们是无法确保它是严格小于 $|t|$ 的. □

4.7 Resolution

Remark 251 这章的目的是证明某个 classical first-logical formula F is unsatisfiable.

Definition 252 The proof of a formula's unsatifiability is called a refutation.

Annotation 253 (formula transformations) 任意一个 F 它的结构可能是多种多样的，我们肯定首先要 format it 得到一个 normalization form. 我们将经历这样的几个过程:

1. 首先把所有的 quantifier 提到最前面 (总是可以做到的)，得到 prenex form $\cdots\exists x_i.\cdots\forall y_j.\cdots F'(\cdots,x_i,\cdots,y_j,\cdots)$，其中 $F'(\cdots,x_i,\cdots,y_j,\cdots)$ 是quantifier-free的.
2. 再把 $F'(\cdots,x_i,\cdots,y_j,\cdots)$ 里面所有的 $\neg$ 放到 atom 前面 (也总是可以做到的)，得到对应的negation form $F''(\cdots,x_i,\cdots,y_j,\cdots)$.
3. 接着把 $F''(\cdots,x_i,\cdots,y_j,\cdots)$ 变成CNF形式，得到 $F'''(\cdots,x_i,\cdots,y_j,\cdots)$
4. 最后去掉所有的 $\exists$ 得到对应的skolemization $\cdots\forall y_j.\cdots F'''(\cdots,f(y_k,\cdots,y_m),\cdots,y_j,\cdots)$

最后我们 generalize 一下

$$F=\forall x_1.\cdots\forall x_n.(L_{11}\vee\cdots\vee L_{1n_1})\wedge\cdots\wedge(L_{k1}\vee\cdots\vee L_{kn_k}).$$

其中 L_{ij} 为一个 literal. 对应的 $\neg F$ 为

$$\begin{aligned}\neg F&=\neg(\forall x_1.\cdots\forall x_n.(L_{11}\vee\cdots\vee L_{1n_1})\wedge\cdots\wedge(L_{k1}\vee\cdots\vee L_{kn_k}))\\&\equiv\exists x_1.\cdots\exists x_n.\neg((L_{11}\vee\cdots\vee L_{1n_1})\wedge\cdots\wedge(L_{k1}\vee\cdots\vee L_{kn_k}))\end{aligned}\tag{1}$$

Theorem 254 (Herbrand's theorem) Given a formula $\exists x_1.\cdots\exists x_n.F(x_1,\cdots,x_n)$, where F is quantifier-free, is valid if and only if there exists a finite set of term $t_{ij}, 1\le i\le k$ and $1\le j\le n$ such that the disjunction:

$$F(t_{11},\cdots,t_{1n})\vee\cdots\vee F(t_{1k},\cdots,t_{nk})$$

is valid.

Annotation 255 Hthe 最关键的作用就是把 first-order logic 转换到 propositional logic. 证明的 Hthe 的 sketch:

1. 首先把它放到 cut-free sequent calculus 里面，根据 cfsc 的 completeness 得到一个 proof(稍显不那么 formal，因为没有 sequent calculus 里面带 $\vdash$vaild 形式):

$$\dfrac{\dfrac{\overset{\mathcal{D}}{\exists x_1.\cdots\exists x_{n-1}.F(x_1,\cdots,x_{n-1},t'_n)}}{\exists x_1.\cdots\exists x_n.F(x_1,\cdots,x_n)}R\exists\quad\dfrac{\overset{\mathcal{E}}{\exists x_1.\cdots\exists x_{n-1}.F(x_1,\cdots,x_{n-1},t''_n)}}{\exists x_1.\cdots\exists x_n.F(x_1,\cdots,x_n)}R\exists}{\exists x_1.\cdots\exists x_n.F(x_1,\cdots,x_n)}\ contraction$$

2. 下一步就是从上往下把 existential quantifier，即去掉 $R\exists$ rule，直接把 primise 换成 conclusion，相当于就是一个 *id* rule. 这里你会发现有些关于 existential quantifier 的 *contraction* rules 也是用不了，因此也要去掉. 最终我们就得到了 Hthe 的结论.

Annotation 256 (from $\neg F$ back to F) 这里设 C_i 表示 $(L_{i1} \vee \cdots \vee L_{in_i})$，对 (1) $\neg F$ 可以 apply Hthe 得到 $\neg F$ is vaild iff

$$\neg(C_i \wedge \cdots \wedge C_k)[\bar{t}_1/\bar{x}] \vee \cdots \vee \neg(C_i \wedge \cdots \wedge C_k)[\bar{t}_m/\bar{x}]$$

is vaild for a finite sequence of term sequences $\bar{t}_1, \cdots, \bar{t}_m$. 比较 trick 是我们需要把它再转回来，证明 $\neg\neg F$ is unsat，此时

$$\begin{aligned} &\neg\left(\neg(C_i \wedge \cdots \wedge C_k)[\bar{t}_1/\bar{x}] \vee \cdots \vee \neg(C_i \wedge \cdots \wedge C_k)[\bar{t}_m/\bar{x}]\right) \\ \equiv &\left((C_i \wedge \cdots \wedge C_k)[\bar{t}_1/\bar{x}] \wedge \cdots \wedge (C_i \wedge \cdots \wedge C_k)[\bar{t}_m/\bar{x}]\right) \end{aligned} \tag{2}$$

Annotation 257 (the context of clauses) 这里给定一下准确的 clause 的定义，后面提到的 clause 的时候都遵循这个定义. clause 里面 literals 使用到的 connective 是 disjunction $\vee$，同时我们说 a set of clauses 的时候，这些 clauses 实际是用 conjunction 连接在一起的. 可以看到前面关于 F 的 normalization form 也是符合这个定义的.

Annotation 258 (another perspective of sat) 在开始讨论后面内容之前，有必要补充一些 extra knowledge, 或者另外一种视角，如果缺失这些东西的时候，会让后面的过程变得有那么一点僵硬? 亦或者你能懂每一步是合理的，但是所有步骤合起来看你会不知道它究竟在干啥. 值得注意是此时我们开始讨论的目标 quantifer-free a set of clause 的 satisfiability，更准确说应该是 unsat，这得力于 Hthe 带来的巨大优势. 那么这里我们可以很自然地从 model 的另外一种定义19，给出关于 satisfiable 的第二种 definition 259. 其中的 Herbrand universe 实际上和 universal algebra 里面的 $Sg(X)$ 是一样的.

那么接下来就是谈谈如何理解这个特殊的 definition，在 classic logic 下的 satisfiable 是说我们总能找到一种 interpretation 让 F is true，也就是给 F 里面的所有 symbols 都给定一个对应的 evaluation，因此当我们对 F 做 evaluation 的时候，最后就可以得到一个 result 为 true.

而此时 satisfiability 变得更加具体了. 对于一个 complementary atoms pair $a \wedge \bar{a}$，这里我们用 $\bar{\bullet}$ 表示 complementary，它显然是 unsat，这就是我们对于 unsat 已经有的 basic ground fact. 如果把 atoms 换成 ground clauses, 同时它也是 unsat，那就意味着无论我们怎样从每个 clause 挑一个 literal 构成的 $a_1 \wedge \cdots \wedge a_n$，总存在 a_i 和 a_j 是 complementary. 当我们考虑所有 clauses 的时候，这时候里面可能就存在 variables，首先需要将它们替换成可能的 terms，得到对应的所有可能的 ground clauses $H(S)$，再考虑利用前面提到的步骤来考虑 unsat. 如果你理解了这里的 unsat 是怎样 processing 的，再看后面 DPM 算法就比较清晰了.

Definition 259 If M is model and S is a set of ground clauses, then M is a model of S if, for all C in S, C contains a member of M. In general, if S is any set of clauses, and H is the Herbrand universe of S, we say that M is a model of S just in case that M is a model of $H(s)$

Definition 260 (Davis-Putnam method)

Algorithm 1: Davis-Putnam method

```
1  begin
2      F ← ∀x̄.F'(x̄)    ▷ F' is CNF and x̄ = (x_1, ··· , x_n)
3      terms → [t̄_1, t̄_2, ···]    ▷ terms is an infinite stream of sequences of terms used for x̄
        in lexicographic order
4      i ← 1
5      G ← F'[t̄_i/x̄]
6      G ← simplify(G)
       ▷ empty clause means that C_i was evaluated to false
7      while G does not contain an empty clause do
8          i ← i + 1
9          G' ← F'[t̄_i/x̄]
10         G ← G ∧ G'
11         G ← simplify(G)
```

The *simplify* method applies the following simplications to the formula:

- **Rule for elimination of single-literal clause**

  1. If G contains a clause $\{p\}$ and other clause $\{\overline{p}\}$, then we are done. They cannot both be true at same time, so the formula is unsatifiable.
  2. If G contains a clause $\{p\}$. we can give a interpretation makes p is false then G is false, but G needs to be false for every interpretation, thats is not a problem. If given p is true under some interpretations, in the case, we can remove the clause $\{p\}$, every clause that contains p (a disjunction with a true literal is true), and remove $\overline{p}$ from every clause that contains it ($\overline{p}$ is the negation of p, that is a false literal in a disjunction is good for nothing).
  3. If G contains a clause $\{\overline{p}\}$, its similar as the case $\{p\}$.

- **Affirmative-Negative rule**
  If p occurs only positively or only negatively in G, then we can delete all clauses containing p without loosing unsatisfiability.

- **Rule for eliminating atomic formulas**

If G contains a clause $\{p\} \cup C_i$ and another clause $\{\overline{p}\} \cup C_j$, then we can replace both of them by the clause $C_i \cup C_j$.

Annotation 261 关于 DPM 的解释:

- 为什么 $G = G \wedge G'$? 主要这里的 F 是限定在 unversal quantifers 下的，如果 F 想要在 unsat，那么对于任意的 substitution instances 在任意的 interpretations 下都应该是 unsat. 同时根据 Hthe，我们知道 F is unsat 当且仅当存在 conjunction of finite substitution instantces is unsat. 这样当 F is unsat 的时候，能保证我们 process 可以停止，如果 F is not unsat，那么这个过程会一直运行下去.
- simplify 详解: simplify 本质就是在不影响原公式 unsatisfiable 的情况下，尽可能的简化 current formula，即the resulting formula is unsat iff the original formula is. 同时 DPM 的最重要的思想就是如果你想证明一串 conjunction $C_1 \wedge C_n$ is unsat, 你只要去保证其中一个 conjunct C_1 is always false 就行了，换句话说我们将忽略某些其他的 conjuncts.
  1. 对于 single-literal clause 的讲解已经比较清楚: 如果存在 $\{p\}$ 和 $\{\overline{p}\}$ 冲突，这种情况很容易想明白; 而对于没有冲突的 $\{p\}$，因为总存在使得 p is true 的 interpretation，那么有 p 存在的 clauses 也都是 true 在这种情况下 F 整体是否为 false 与它们没有关系，因此去掉它们并不会影响 F is unsatifiable. 注意此时 $\overline{p}$ is false，对于它所在的 clause 是否 false 和它也没有关系，除非这个 clause 只含有 $\overline{p}$，那又回到第一种情况了，因此可以去掉这样的 $\overline{p}$. 其余使得 p is false 的 interpretations，直接就让 F 变成 false，因此这种情况不用考虑了. 仔细体会其中的奥义，我们在渐渐的逼近那个 always false 的 clause，prune 掉所有不会影响 F is unsat 的 clause. 在这种情况下它们与 F 整体是否为 false 与 p 没有关系.
  2. 例如假设只有 p 存在 (没有 $\neg p$ 存在)，因为总存在使得 p is true 的 interpretations，使得 p 所在 clauses 也是都是 true，在这种情况下整体 F 是否为 false，实际上与它没有关系. 同理只存在 $\neg p$ 的情况也是一样的.
  3. 对于任意的 interpretation 都可以分成两种情况: 当 p is true，the conjunction of clause is false 取决于 C_j，如果 C_j is empty clause，那么整体就是 false; 当 p is false, the conjunction clause is false 取决于 C_i，如果 C_i is empty clause. 那么只有 $C_1 \cup C_2$ 是 empty clause 的情况下整体才是 false，因此我们可以 safely replace both of them with $C_1 \cup C_2$.
- 为什么 while 的循环条件是不包含 *empty clause*? 从 simplify methods 中你可以看到 clauses and literals those not affect unsat will be removed，显然出现 empty clause 就是已经 unsat 了.

作为 SAT slover 基础的 DPPLL algorithm 就是 refined from DPM.

Example 262 如果给定下面的 formula

$$F = \forall x. \forall y. (P(x) \wedge \neg P(f^{100}(y)))$$

显然是 unsat，例如当 $x = f^{100}(a), y = a$ 时，F is false. 但是应用 DPM 可以就会进入漫长的 tests，例如 terms will be $[(a,a),(f(a),a),\cdots]$.

Annotation 263 (Why resolution ?) Example 262指出 DPM 存在的一个显著的问题或者遗留的问题，就是 how to generate terms systematiclly?，此外我们也希望 terms is efficient for our tests! 由此 Resolution 就提出来了.

Definition 264 If C_1 and C_2 are two ground clauses, and $L_1 \subseteq C_1, L_2 \subseteq C_2$ are single literal whose respective members form a complementary pair, then the ground clause $(C_1 - L_1) \cup (C_2 - L_2)$ is called a ground resolvent of C_1 and C_2.

Annotation 265 在 definition 264下，显然任意的 model of $\{C_1, C_2\}$ 也是 model of $\{C_1, C_2, R\}$, 其中 R 为 $(C_1 - L_1) \cup (C_2 - L_2)$. 这是因为根据 model 定义，设 M 是 $\{C_1, C_2\}$ 的一个 model, 那么 M 一定不会是 $\{L_1, L_2\}$, 因此一定存在 $L \in M$ such that R contains it，同时 M 也不会存在 complementary pair，因此 M 也是 R 的 model. 前面这个结论反过来，显然也是正确的. 上面结论是比较重要的，按照下述关于 $\mathcal{R}(S)$ 的定义，我们可以给出它 generalization: 任意的 model of the set of ground clauses S 也是 model of $\mathcal{R}^n(S)$

Definition 266 If S is any set of ground clauses, the ground resolution of S ,denoted by $\mathcal{R}(S)$, is the set of ground clauses consisting of the ground literals of S together with all ground reslovents of all possible pairs of ground literals of S. Then we can define nth ground resolution of S, denoted by $\mathcal{R}^n(S)$, is defined for each $n \geq 0$ as follows: $\mathcal{R}^0(S) = S$; and for $n \geq 0$, $\mathcal{R}^{n+1} = \mathcal{R}(\mathcal{R}^n(S))$.

Theorem 267 [14] (ground resolution theorem) If S is any finite set of ground clauses, then S is unsatisfiable if and only if $\mathcal{R}^n(S)$ containts $\square$, for some $n \geq 0$.

Proof $\Rightarrow$ 这个方向还是比较明显的，通过265里面的提到结论就可以证明. 对于 $\Leftarrow$ 方向我们可以证明当 $\mathcal{R}^n$ 不含有 $\square$ 的时候，S 是 satisfiable 的，相当于原命题的逆否命题. 这里原论文中用了一个巧妙的构造 model 的方法，我会尽可能将其描述的更详细一些. 由于 S is finite, 那么关于 $\mathcal{R}^n$ 的构造一定会 terminate 在某个地方，我们设为 $\mathcal{R}^k$，记它为 T. 接着我们设 T 所有的 distinct atoms 为 $L_1, \cdots, L_m$, 注意这里描述的 distinct atoms 并不是指得是 T 里面的 distinct literals，也就是说如果 a 和 $\bar{a}$ 同时在 T 里面或者只有 $\{a\}$ 在 T 里面，那么只有 a 作为 atom 出现在前面的 sequence 里面，是没有它的 complementary 的.

我们构造 model M 如下:

- $M_0 = \emptyset$.
- 对于 $j \geq 1$, (1)如果存在 $C \in T$ 满足 C consists of some complements of literals in $M_{j-1} \cup \{L_j\}$，那么 $M_j = M_{j-1} \cup \overline{L_j}$; (2) otherwise $M_j = M_{j-1} \cup L_j$.

最后使得 $M = M_k$. 接下来我们需要说明在 T 不含 $\square$ 的情况下, M is model of T.

这个构造方式中条件关系，感觉非常不符合直觉，就是 if A, then B, 但是 A 和 B 之间好像没什么联系??? 这是理解这个构造方式最难的地方. 当我们手推 M_k 时，就可以抓住这里面微妙的东西. 当 $j = 1$ 时，$M_0 \cup \{L_1\} = \{L_1\}$，我们需要 check 一下 T 里面有没有 clause $\{\overline{L_1}\}$，如果有，那么 $M_1 = \{\overline{L_1}\}$, otherwise $M_1 = \{L_1\}$. 接着当 $j = 2, 3, \cdots$ 依次构造，我们可以发现因为 $L_1, \cdots, L_k$ are distinct atoms, 直觉上意味着命中条件(1)的时候，$\overline{L_j}$ 总是在 C 中，但是并不正确. 存在一种情况当 $M_j = M_{j-1} \cup \{\overline{L_i}\}$, 同时存在 clause D contains L_i, 并且 D 里面其他的 literals 也都是 M_{j-1} 里面 literals 的 complement. 这样在进行 $j+1$ 构造的时候，D 可以命中条件(1)，但是 $\overline{L_{j+1}} \notin D$，那么问题来了，一旦出现这样的情况，$D$ 中所有 literals 的 complements 都在 M 里面，那就意味 M 无法 cover 住 D，并且 $j+1$ 之后所有构造 D 都会命中条件(1).

回到问题的核心，假设这样的 M is not model for T, 那么只有在 M 没有完全 cover 到 T 或者出现 complementary literals 的时候. 显然这样构造出来 M 是不会出现 complementary literals 的，因为每次加进去的都是 distinct atoms. 因此我们只需要考虑第一种情况，设 M 没有 cover 某个 clause C', 由于 M cover 到了所有的 distinct atoms 了，其中也包括 C' 中的 atoms，那就意味着 C' 的 complements 全在 M 中，那意味着 C' 会在某个时刻命中条件(1). 经过上述分析，命中条件(1)有两种情况分别对应了 C 和 D，显然不会是 C，因为 M 包含了 C 里面的一个元素，那么只能是 D. 前面提到一旦出现情况 D，后面会总是命中，我们前面描述是 D 第一次命中的时候. 现在 everything is clear now, 我们开始正式来证明 M is model of T: 假设 M is not model for T, 则存在上述提到 clause D. 我们设第一次命中 D 的时刻为 $i+1$, 我们来考虑时刻 i，那么 C 和 D 的 ground resolvent

$$R = (C - \overline{L}_i) \cup (D - L_i)$$

显然 $R \in T$，并且 $\square \notin T$，那么 $R \neq \square$. 注意 R 中的所有 literals 的 complements 都在 M_{i-1} 中，显然 R 可以作为 D 在 i 时刻就命中，这和我们的假设是矛盾的，因此不存在这样的 D. 最终 M is model of T，而 $S \subseteq T$，所以 M is model of S. $\square$

Theorem 268 (ground resolution theorem v2) If S is any finite set of clauses, then S is unsatisfiable if and only if, for some finite subset P of the Herbrand universe of S, and some $n \geq 0$, $\mathcal{R}^n(P(S))$ contains $\square$.

Annotation 269 接下来我们需要 generalize ground resolution，让 S be any clauses，首先得给出对应的 $\mathcal{R}(S)$ 和 $\mathcal{R}^n(S)$ 的 definition，其中关键的是 whats the resolution of S? 这里又要用到 unification 了，它真的是无处不在，我们将记录这里特殊之处. 这里的 unifier 会首先构造下面定义的 disagreement set, 从里面取 terms 出来做 unification. 我们说 a set of well-formed expressions W is unifiable, 当且仅当存在 substitution σ 使得 $W\sigma$ is singleton.

Definition 270 The Pdisagreement set for a set W of well-formed expressions is obtained by finding the first position (staring form the left) at which not all expressions in W have the same symbol. We then extract from each expression in W the subexpression that begins with the symbol occupying that position. The set if these sub-expression is the disagreement set for W.

Example 271

$$A = \{P(x, h(x, y), y), P(x, k(y), y), P(x, a, b)\}$$

那么 A 的 disagreement set 为 $\{h(x, y), k(y), a\}$.

Definition 272 If C is any finite set of strings, and $v_1, \cdots, v_n$ are all the distinct variables, in alphabetical order, which occur in strings in C, then x-standardization of C, denoted by τ_x, is the substitution $[v_1 \to x_1, \cdots, v_n \to x_n]$, and similarly the y-standardization of C, denoted by τ_y, is the substitution $[v_1 \to y_1, \cdots, v_n \to y_n]$.

Definition 273 A resolvent of the two clauses C_1 and C_2 is any clause of the form $((C_1 - L_1) \cup (C_2 - L_2))\sigma_N$ such that satisfies following conditions:

- L_1 and L_2 are nonempty, and $L_1 \subseteq C_1, L_2 \subseteq C_2$.
- N is the set of atoms which are literals, or complements of literals of the set $(L_1\tau_x) \cup (L_2\tau_y)$(why need standardization?).
- N is most generally unifiable, with most general unifier σ_N.
- The sets $L_1\tau_x\sigma_N$ and $L_2\tau_y\sigma_N$ are singletons whose literals are complentary.

And we call $\langle C_1, C_2, N\rangle$ is a key triple of pair (C, D).

Annotation 274 这里我们需要特别解释一下 definition 273里面的 L_1 和 L_2 要做standardization? 是因为你不做，后面的 resolution 就不够 strong，这一点困扰了我非常久，最终在 [15] 里面找到了答案. 它里面用到一个例子，如果给定 S 为 $\{P(a, x), \neg P(x, b)\}$, 你如果不用 variables standardization, 显然 S 里面这两个 atoms 是无法做 unification 的，但是 S is unsat, i.e., two instances $P(a, b) \wedge \neg P(b, b)$ $(x = b)$ 和 $P(a, a) \wedge \neg P(a, b)$ $(x = a)$. 后面我们希望的是 resolution 足够 strong，即如果 S is unsat, then we can get empty clause $\square$ from resolution of S. 幸运地是 variables standardization 刚好在这里帮助我们解决了这个问题.

特别地，再解释一下 N 的取法，直觉上我们希望的是这样的过程

$$\frac{C_i \cup \{p\} \quad C_i \cup \{\overline{p'}\}}{(C_i \cup C_j)\sigma} \ \textit{resolution}$$

其中 σ 是关于 p 和 p' 的 most general unifier. 但是 N 却直接取了一堆 atoms 出来，这里面实际上蕴含这样一个过程

$$\frac{C_i \cup \{p, p'\}}{(C_i \cup \{p\})\sigma} \ \textit{factoring}$$

其中 σ 是关于 p 和 p' 的 most general unifier. 所以这里不仅在 unify may complementary literal, 同时也在 unify may equivalent literals. 聪明的你一定还会发现这里应该还有一个取法

$$\frac{C_i \cup \{p, \overline{p'}\}}{?}$$

其中存在一个 σ 为 p 和 p' 的 most general unifer，简单想一下如果出现这样的情况，那么 $C_i \cup \{p, \overline{p'}$ 本身就是 satisfiable，这个 clause 直接可以去掉了，已经不在 resolution 的范畴里面了.

Theorem 275 Resolution Theorem If S is any finite set of clause, then S is unsatisfiable if and only if $\mathcal{R}^n(S)$ contains $\square$, for some $n \geq 0$.

Annotation 276 (The story of resolution theorem) 这就是真正的"A complete system is formed by one inference principle" . 关于上面 resolution theorem 推出实际上还有一些前奏，即它是怎样最终归纳到 ground resolution theorem 上的，这里就不得不提两个东西了:

1. $P(\mathcal{R}^n(S))$: the finite set of Herbrand universe of resolution of S.
2. $\mathcal{R}^n(P(S))$: the resolution of set of finite set of Herbrand universe of S.

其中 S 都是 the finite set of clauses. 这两个作为 ground clauses 都是 vaild 的，但是 $|mathcal R^n(P(S))| \leq |P(\mathcal{R}^n(S))|$. 因此这里有一个前奏 Theorem，就是把 resolution theorem 中的 $\mathcal{R}^n(S)$ 换成 $P(\mathcal{R}^n(S))$，换言之 resolution theorem 只不过是它的 simplification.

Example 277 证明: 若给定一个 subgroupp $(S, \cdot)$, 对任意的 $a, b \in S$, 总存在 x, y 使得 $x \cdot a = b$ and $a \cdot y = b$. 则 $(S, \cdot)$ is monoid, 换句话说就是存在一个 identity element. 常规证明思路: 对任意两个元素 $x \in S, y \in S$，设 e 满足 $e \cdot x = x$，利用 extension of right 存在 z 使得 $x \cdot z = y$. 再用一下 associativity:

$$e \cdot (x \cdot z) = e \cdot y$$
$$(e \cdot x) \cdot z = y$$

就可以得到 $e \cdot y = y$，即 x 的 right identify 也是 y 的 identity，感觉像是一个很弱的条件得到了一个很强的条件 xd. 我们可以尝试用 resolution 来证明它，这里只试着证明 left identity element，首先需要把已知的 knowledges 都列出来. 这里用 $\sim$ 来表示 complement，感觉比较方便. 我们用 $Q(x, y, z)$ 表示 $x \cdot y = z$:

$$\begin{aligned}
&C_1 : \{Q(x,y,u) \wedge Q(y,z,v) \wedge Q(x,v,w) \to Q(u,z,w)\} && \text{associativity} \\
&C_2 : \{Q(x,y,u) \wedge Q(y,z,v) \wedge Q(u,z,w) \to Q(x,v,w)\} && \text{associativity} \\
&C_3 : \{Q(g(x,y),x,y)\} && \text{existence of left} \\
&C_4 : \{Q(x,h(x,y),y)\} && \text{existence of right} \\
&C_5 : \{Q(x,y,f(x,y))\} && \text{closed under } \cdot
\end{aligned}$$

其中 C_1 和 C_2 实际来于对 equivalention 的 transformation，这里省略了，接着还需要做一下 transformation 去掉 implication:

$$C_1 : \{\sim Q(x,y,u), \sim Q(y,z,v), \sim Q(x,v,w), Q(u,z,w)\} \quad P \to Q \iff \neg P \vee Q$$
$$C_2 : \{\sim Q(x,y,u), \sim Q(y,z,v), \sim Q(u,z,w), Q(x,v,w)\}$$

为了能构造 refutation，我们还得把 conclusion $\exists e.\forall x.\ Q(e,x,x)$ 的 negation $\forall e, \exists x.\ \neg Q(e,x,x)$ 放到里面去，这里还需要再转一下 Skolemization:

$$C_6 : \{\sim Q(e, k(e), k(e))\}$$

我们可以构造一个 refutation 如下

$$\begin{aligned} C_7 &: \{\sim Q(x_4,y_1,y_1), Q(x_4,y_2,y_3)\} && \text{resolvent of } \{C_2, C_4\} \\ C_8 &: \{\sim Q(y_1,y_2,y_2)\} && \text{resolvent of } \{C_6, C_7\} \\ C_9 &: \square && \text{resolvent of } \{C_3, C_8\} \end{aligned}$$

其中 C_7 中 $L_1 = \{\sim Q(y,z,v), \sim Q(u,z,w)\}, L_2 = \{Q(x,h(x,y),y)\}$，因此

$$N = \{\sim Q(x_5,x_6,x_2), \sim Q(x_1,x_6,x_3), Q(y_1,h(y_1,y_2),y_2)\},$$

那么这里 $\sigma_N = [x_1 \to y_1, x_2 \to y_2, x_3 \to y_2, x_5 \to y_1, x_6 \to h(y_1,y_2)]$. 至于 C_8 和 C_9 都比较明显，从这个例子里面你应该可以看见 automated theorem proving is possible.

Annotation 278 虽然我们丢掉了 ground clauses 的生成，但是问题又来了 $\mathcal{R}^n(S)$ 你又得想 Herbrand universe 一样算, XD. 但是无论如何至少少了一个 P 的围绕. Robinson 的论文里面也接着有提到几个针对简化手法.

Definition 279 If S is any finite set of clauses, C a clause in S, and L a literal in C with the property that there is no key triple for any pair (C, D) of clauses, where D is any clause in $S - \{C\}$, then L is said to be pure in S.

Theorem 280 (purity theorem) If S is any finite set of clauses, and $L \in C$ is literal which is pure in S, where $C \in S$, then S is satifiable if and only if $S - \{C\}$ is satisfiable.

Definition 281 If C and D are two distinct nonempty clauses, we say that C subsumes D just in case there is a substitution σ such that $C\sigma \subseteq D$.

Definition 282 (subsumption theorem) If S is any finite set of clauses, and D is any clause in S which is subsumed by some clauses in $S - \{D\}$, then S is satisfiable if and only if $S - \{D\}$ is satisfiable.

4.8 SLD Resolution

Annotation 283 SLD 表示 selective linear definite clause resolution，它是对原本 resolution 的一种 refinement，用于 the implementation of Prolog's engine. 这就是为什么这个大章节里面会出现 resolution 的概念.

Definition 284 If there is only one positive literal (atom) in clause C, then C is definite clause.

Annotation 285 首先回忆一下 Horn clauses 长下面这样

$$B_1 \wedge \cdots \wedge B_n \supset H$$

我们将其转换为 CNF 的形式

$$\neg B_1 \vee \cdots \vee \neg B_n \vee H$$

那么它正好就是一个 definite clause. 在 Prolog 里面 make query G，实际上就是要证明 G is vaild under given rules，即 $F = R_1 \wedge \cdots \wedge R_k \subset G$，其中 R_i 表示一段 Prolog program 里面所有 Horn clauses. 也就是要证明 $\neg F$ is unsatifiable, so F gonna to be $R_1 \wedge \cdots \wedge R_k \wedge \neg G$, 它也是下述 clauses:

$$\begin{aligned}
&C_1 : \{\neg B_1^1, \cdots, \neg B_{n_1}^1, H_1\} \\
&C_1 : \{\neg B_1^2, \cdots, \neg B_{n_2}^2, H_2\} \\
&\vdots \\
&C_k : \{\neg B_1^k, \cdots, \neg B_{n_k}^k, H_k\} \\
&C_{k+1} : \{\neg G\}
\end{aligned}$$

SLD 里面规定 resolvents 的产生必须来自两个特别的 clauses C_1 和 C_2，其中 C_1 满足里面全部 literals 都是 negative, 而 C_2 是一个 finite clause. 这样就是总是拿手上的 goals 去 match possible heads in Horn clauses，也正是我们之前谈到的 backing chaining.

Annotation 286 再 buidling linear logic 过程，也许你可以感受一种非常独特看待 theory 的视角. 我们在学习某个 theory 的时候，往往会通过 practices 来指导我们来理解它里面的一些 definitions，但是你如果不能很好的掌握全局话，这样做很难让你继续下去. 它给我带来了一种全新的视角，如果在最开始我们无法用很具体的东西来理解抽象的 definitions，没关系你试着接着走下去，只要这些 definitions 让你感觉舒适就行了，大体上没什么太过怪异的就行，在后面应用这个 theory 的时候，你自然会得到你想要的 interpretations. 带着特殊的 practices 反而会太局限!

5 Linear Logic

Remark 287 *Linear logic is a refinement of classical and intuitionistic logic. Instead of emphasizing truth, as in classical logic, or proof, as in intuitionistic logic, linear logic emphasizes the role of formulas as resources.*

5.1 MALL

Definition 288 A structural rule is an inference rule that does not refer to any *logicial connective*.

Annotation 289 (初见 linear logic) 像我们之前见到几种操作: exchange (把 judgement 的某一端里面的成员交位置), contraction, weakening 这些都属于 structural rules. 后面我们即将提到的 linear logic 就和它们就有着密切的关系，linear logic 关注的对象是 resources，或者更准确的说是 resource availability. 现在假设拥有两个相同的 resources 和只拥有一个 resource，在语境上是不一样的，这里 nlab 上用了一个形象的例子，假如我们有这样的 sequent

$$\text{have cake} \vdash \text{eat cake}$$

那么我们就可以证明

$$\text{have cake}, \text{have cake} \vdash \text{have cake} \wedge \text{eat cake}$$

如果你接着用一下 contraction，语境就不正确了:

$$\text{have cake} \vdash \text{have cake} \wedge \text{eat cake}$$

我们不能在仅拥有一块蛋糕的情况下，选择吃掉了它之后还剩下一块蛋糕，这显然是不正确的. 而在 classic and intuitionistic logic 下有一种与之完全对立的语境:

$$\text{if } A \text{ and } A \supset B, \text{ then } B, \text{ but } A \text{ still holds.}$$

可以看见两者在对 premises 处理上是不一样的，前者需要 “消耗” premises(注意它和 discharge 是不一样的，discharge 本质上一种 identity)，而后者可以依然保持 premises.

Definition 290 (Logical linearity) A proof of $\Gamma, A \to B$ is linear in A if the A is not involved in a contraction or weakning any where in the proof.

Definition 291 we define $\multimap$ for linear implicaction.

Annotation 292 (如何用 linear implication) 我们就可以用 linear implication 来描述上面的 logical linearity，其中 A 表示 have cave，B 表示 eat cake.

$$\text{have cake} \multimap \text{eat cake}$$

那么 A 在上面这个特殊"sequent" 下 ($A \vdash B$ 和 $\vdash A \supset B$ 是等价的, 我们经常说 $A \supset B$ is valid, 就是指 $\vdash A \supset B$ is valid) 下就是 linear 的. 同时从前面的过程我们知道 resources 是可以计数的, 例如有两块蛋糕之后我们可以选择吃掉一块和留下一块. 那么试想如果我们吃不完的蛋糕我们应当如何描述它呢? 难道要用 $A, A, \cdots$ 吗? 我们有更优雅的方式 $!A$.

Definition 293 There are two connectives ! and ? are called exponentials which shall express the iterability of an formula. That is we can use these mark the formulas that be used an unbounded number of times (including zero). ! marks such formulas on the left and ? marks such formulas on the right.

Annotation 294 (linear implication 的等价形式) 现在我们可以给出之前 implication in classical and intuitionistic 和 linear implication 的关系

$$A \subset B \equiv !A \subset B$$

Annotation 295 (通过 sequent calculus 构建 linear logic) 再了解了前面几个关于 linear logic 的有趣的 properties 之后, 我们来正式构建它. 我们知道 classic and intuitionistic logic 的 deduction 都可以依赖 sequent calculus, 那么我们 linear logic 是否也可以依赖 sequent calculus 大体框架呢? 这里面有几个需要思考的问题是:

- contraction 和 weakening rules 如何处理?
- 除了我们前面引入新的 linear implication, 其他的 connective 是否有足够的 expressive 来作为 linear logic 的 presentation?

最初 Gentzen 提出了第一个 sequent calculus LK system, 在它的基础上把 right side 严格限制为了 single formula, 就得到了专为 intuitionistic logic 设计的LJ system. 但是你仔细对比它们在 wiki 上的样子和我们之前在 proof searching 那一章我们介绍的 intuitionistic sequent calculus 是存在差异的, 这是也是曾困扰我的一个地方, 其中的差异体现在部分的 rules, 例如 $\wedge L$. 后来经过查阅相关的资料终于在 [16] 中找到了为什么, 原来这是 Gentzen 的学生 Ketonen 做的一个重要的工作, 也就是我们前面提到的 invertibility. 为什么要介绍这些前因后果, 是因为我们又需要回到最初的 sequent calculus 了, 并且扩充一下它的另外一种形式. 我们之前看到的 sequent calculus inference rules 全都是 *additive form*, 还存在另外一种在 classic logic 和 intuitionistic logic 下等价的形式 *multiplicative form*, 相比于 *additive form* 它具有更弱一些 premises, 下面给出它们的对比:

Additive	Multiplicative
$\dfrac{\Gamma \to A \quad \Gamma \to B}{\Gamma \to A \wedge B}\ \wedge R$	$\dfrac{\Gamma_1 \to A \quad \Gamma_2 \to B}{\Gamma_1, \Gamma_2 \to A \wedge B}\ \wedge R$
$\dfrac{\Gamma, A_i \to C}{\Gamma, A_1 \wedge A_2 \to C}\ \wedge L$	$\dfrac{\Gamma, A_1, A_2 \to C}{\Gamma, A_1 \wedge A_2 \to C}\ \wedge L$
$\dfrac{\Gamma \to A_i}{\Gamma \to A_1 \vee A_2}\ \vee R$	$\dfrac{\Gamma \to A_1, A_2}{\Gamma \to A_1 \vee A_2}\ \vee R$
$\dfrac{\Gamma, A \to C \quad \Gamma, B \to C}{\Gamma, A \vee B \to C}\ \wedge L$	$\dfrac{\Gamma_1, A \to C \quad \Gamma_2, B \to C}{\Gamma_1, \Gamma_2, A \vee B \to C}\ \wedge L$

删掉 $\vee R$ 的原因是是它在 intuitionistic logic 中是不被允许的. 这里我们还是重点记录一下 intuitionistic linear logic. 引出这两种 forms 是为了让我们在 sequent calculus 的基础上提出新的 connectives 变成可能. 我们之前在构造新 logic 的时候, 并没有改变 operators, 而是在 rules 上做一些技术性的操作. linear logic 就不太一样了, 我们会提出新的 connectives, 使其更加具有 expressive.

Definition 296 The additive and multiplicative forms of conjunction and disjunction are defined as follow

	Additive	Multiplicative
Conjunction	&(with)	$\otimes$(time)
Disjunction	$\oplus$(plus)	$\parr$ (par)

Annotation 297 (关于新的 connectives 的简单解释) 这里我们解释两个名词 action 和 type 为了后面更好的说明:

- action 实际上就是我们之前所说的 instance, 只不过把它用在对 resources 上 formulation 会显得比较合适.
- type 就是由 scheme variables 组成的 formula, 例如 $A \,\&\, B, A \otimes B$. 你用 Curry-Howard ismorphism 来看就比较能理解了, 某个 action 就是具有某种 type 的 instance.

首先我们来看 additive conjunction & 和 multiplicative conjunction $\otimes$, 它们都可以表示 availability of two actions, 不同的是 $\otimes$ 强调是两个 actions will be done, & 强调最终只有一个 action will be done. 例如 A 表示 have cake, B_1 和 B_2 分表表示 maple eats cake 和 laura eats cake, 那么对于

$$A \multimap B_1 \,\&\, B_2$$

是没问题的, 但是 $A \multimap B_1 \otimes B_2$ 显然就不行了. 我们再来看一下 additive disjunction $\oplus$ 和 multiplicative disjunction $\parr$. 其中 $\otimes$ 可以表示 the choice of one action between two possible types, 这里它仅仅只是表示一种 choice, 并不是和 & 里面那样选了之后还要 be done. 而 $\parr$ 可以表示 the dependency between two types of actions.

Definition 298 The additive and multiplicative rules of conjunction and disconjunction of linear logic as follow:

Additive	Multiplicative
$\dfrac{\Gamma \to A \quad \Gamma \to B}{\Gamma \to A \,\&\, B}\ \&R$	$\dfrac{\Gamma_1 \to A \quad \Gamma_2 \to B}{\Gamma_1, \Gamma_2 \to A \otimes B}\ \otimes R$
$\dfrac{\Gamma, A_i \to C}{\Gamma, A_1 \,\&\, A_2 \to C}\ \&L$	$\dfrac{\Gamma, A_1, A_2 \to C}{\Gamma, A_1 \otimes A_2 \to C}\ \otimes L$
$\dfrac{\Gamma \to A_i}{\Gamma \to A_1 \oplus A_2}\ \oplus R$	$\dfrac{\Gamma \to A_1, A_2}{\Gamma \to A_1 \parr A_2}\ \parr R$
$\dfrac{\Gamma, A \to C \quad \Gamma, B \to C}{\Gamma, A \oplus B \to C}\ \oplus L$	$\dfrac{\Gamma_1, A \to C \quad \Gamma_2, B \to C}{\Gamma_1, \Gamma_2, A \parr B \to C}\ \parr L$

Annotation 299 (linear logic 似乎对应某个 type system) 似乎 $\otimes$ 和 sum type 对应上了，上面的 inference 非常像 match syntax, 但是 $\otimes$ 是不是和 production type 对应上了，还不太确定.

Definition 300 Different connective require different neutral elements:

	Neutral "true"	Neutral "false"
Additive	$\top(\&)$	$0(\oplus)$
Multiplicative	$1(\otimes)$	$\bot\ (\parr)$

Definition 301 The rules for neutral elements:

$$\frac{}{\Gamma \to \top}\ \top R$$

$$\frac{}{\cdot \to 1}\ 1R \qquad \frac{\Gamma \to C}{\Gamma, 1 \to C}\ 1L$$

$$\frac{}{\Gamma, 0 \to C}\ 0L$$

Annotation 302 (对 netural elements 的思考) 关于 neutral elements 的 rules 你一定会非常的疑惑，它的存在就为了对应 classical logic 里面的 truth 和 falsehood，这里多了一对 conjunction 和 disjunction，自然有两对 truth-falsehood. 先简单地解释一下关于这些 netural rule 的构造，这些 neutral elements 都看 zero conjuncts 或者 zero disjuncts，我们再看左右两边都是如何 introduce 它们的，整个下来就比较清晰了. 例如 & 的 $\top$, 如果 $\top$ 在 right side，根据 $\&R$，它就就需要 conjuncts with some contexts 作为 primise，但是此时 zero conjuncts，因此是 empty primise; 如果 $\top$ 在 left side，根据 $\&L$，我们需要从 $\top$ 选一个 conjunct 出来，但是 $\top$ is zero conjuncts，因此没有 $\top L$. 如果你想既然 $\top$ 是 zero conjuncts, 那么为何不能这样呢

$$\frac{\Gamma \to C}{\Gamma, \top \to C}\ \top L$$

如果有它存在，那么我们会得到

$$\dfrac{\dfrac{\dfrac{\Gamma \to C}{\Gamma, \top \to C}\,\top L}{\Gamma, \top \,\&\, A \to C}\,\&L_1}{\Gamma, A \to C}\,neu$$

这下把 weakeaning 给造出来了 XDDD.

Annotation 303 (对于 $\multimap L$ shape 的解释) 下面 linear implication 和 sequent calculus 的 implication 是相似的，这里的 $\multimap L$ 也是用最开始 non-invertiable $\supset L$ 模子.

Definition 304 The ruls for linear implication, denoted by $\multimap$, can be written as follow:

$$\dfrac{\Gamma, A \to B}{\Gamma \to A \multimap B}\multimap R \qquad \dfrac{\Gamma_1 \to A \quad \Gamma_2, B \to C}{\Gamma_1, \Gamma_2, A \multimap B \to C}\multimap L$$

Annotation 305 (linear implication 的逻辑等价形式) 实际上 MALL 里面是没有 linear implication 这个 connective，而是通过引入 negation 来做一个 logical equivalent

$$A \multimap B \equiv \neg A \parr B$$

你可以通过上述的 linear implication rules 来证明一下:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{A \to A}\,init \quad \dfrac{}{B \to B}\,init}{A \multimap B, A \to B}}{A \multimap B \to \neg A, B}\,neg\,R}{A \multimap B \to \neg A \parr B}\parr R$$

反过来也一样. 注意这个 linear implication 只能出现在 classical linear logic 里面，不在 intuitionistic linear logic 里面（无法理解其中的意思）.

Definition 306 The initial rule as follow

$$\dfrac{}{a \to a}\,init$$

Annotation 307 这个 initial rule 非常自然地描述了 resources availability.

Definition 308 The structural rules of exponentials as follow:

$$\dfrac{\Gamma, !A, !A \to C}{\Gamma, !A \to C}\,!cL \qquad \dfrac{\Gamma \to C}{\Gamma, !A \to C}\,!wL \qquad \cancel{\dfrac{\Gamma \to ?A, ?A}{\Gamma \to ?A}\,?cR} \qquad \cancel{\dfrac{\Gamma \to}{\Gamma \to ?A}\,?wR},$$

Definition 309 The logical rules of exponential ! as follow:

$$\dfrac{\Gamma, A \to C}{\Gamma, !A \to C}\,!L \qquad \dfrac{!\Gamma \to A}{!\Gamma \to !A}\,!R$$

Annotation 310 其中 rule $!L$ 是比较好理解的 "finite resource A can prove C, then unbounded resource $!A$ also can." 如果 $!A$ 出现来右边，它可以解释为"A can be proved with unbouned resources, then $!A$ also can be proved".

Definition 311 The logic composed of the connectives seen so far (including $\parr$ and $\bot$) constitutes of the multiplicative-additive fragment of linear logic, called MALL.

Annotation 312 总结

5.2 States and Transitions

Annotation 313 在 Kripke frame 里面我们提到过关于 finite state machine(FSM) 的 reasoning，但需要构造几个特殊的 modalities 才行，不过还是比较自然的. 在 FSM 上做 reasoning 之前，你得先想办法 encode 它，其中包括三个重要的部分: initial state, states, transitions; 如果你考虑 classi logical 或者 intuitionistic logical 来做 encoding 的话，面临困难是处理 transitions，因为 states 的更新不能用通常意义下的 implication 来完成，但是你考虑 linear implication 时候就比较合适了. 例如 $\multimap L$:

$$\frac{\Gamma_1 \to A \quad \Gamma_2, B \to C}{\Gamma_1, \Gamma_2, A \multimap B \to C} \multimap L$$

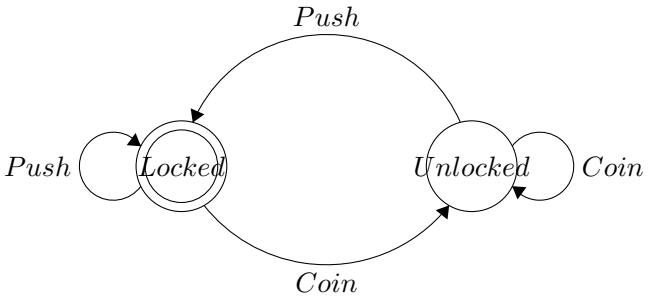
如果 $A \multimap B$ 表示一个 transition，那么上面两个 primises 是非常好解释的: 左边 $\Gamma_1 \to A$ 表示某个 A is available in Γ_1, 而右边我们可以得到 Γ_2 with B is available and A is not available, 这样很自然的表示了 update. 其次 $\otimes$ 的 expressive 也是在 state representation 中比较重要的，例如你如果考虑是化学反应状态变化，在某个状态下其中化合物比例是必须记录的量，这个时候就需要 $\otimes$ 来发挥作用了.

Definition 314 The logical representation of finite state machine is defined as follow:

$$\text{FA} = \text{linear logic} \ + \ \text{axioms} \ + \ \text{initial state}$$

where aximos can of course be replaced by $!A$.

Example 315 设我们下面这样一个 FSM



根据上图我们可以得到下面的 axioms:

- $T_1 = locked \otimes coin \multimap unlocked.$
- $T_2 = locked \otimes push \multimap locked.$
- $T_3 = unlocked \otimes coin \multimap unlocked \otimes coin$ (注意这里我们保留了 coin).
- $T_4 = unlocked \otimes push \multimap locked$

那么我们可以来试着证明一下 $coin \multimap unlocked$:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{locked \to locked}\ init \quad \dfrac{}{coin \to coin}\ init}{locked, coin \to locked \otimes coin}\otimes R \quad \dfrac{}{!T_2, !T_3, !T_4, unlocked \to unlocked}\ !wL^* + init}{locked, T_1, !T_2, !T_3, !T_4, coin \to unlocked}\multimap L}{locked, !T_1, !T_2, !T_3, !T_4, coin \to unlocked}\ !L}{locked, !T_1, !T_2, !T_3, !T_4 \to coin \multimap unlocked}\multimap R$$

5.3 Focused Linear logic

6 More Delicate

6.1 Box is Powerful

Definition 316 $\Box$ is $\Box$.

Definition 317 A term box$M$ means M is a quated source expression such that there are not any free variables x.

Definition 318 And $\Box A$ is necessity modality.

6.2 Validity

Definition 319 A *valid* if $\bullet \vdash A$ *true* where $\bullet$ is emphasizing that there are no truth hypotheses(different from $\cdot$ that represents empty collection of hypotheses), and we call $\bullet \vdash A$ *true* is categorical judgement. Written ΔA for reflecting the notion of validity as a proposition.

Annotation 320 其中 $\Box A$ 表示一个 proposition claimed A is vaild, 因此 $\Box A$ *true* 表示这个 proposition 成立. 那么关于它的 introduction rule 是什么? 很自然地由 A *valid* 的 definition 有

$$\frac{\bullet \vdash A\ true}{\Gamma \vdash \Box A\ true}\ \Box I$$

那么它的 elimination rule 又是什么呢? 第一次尝试

$$\frac{\Gamma \vdash \Box A\ true}{\bullet \vdash A\ true}\ \Box E$$

看起来是 local soundness，通过它得到的 infos 还行. 但是实际上有问题

$$\frac{\overline{\Box A\ true \vdash \Box A\ true}}{\bullet \vdash A\ true}\ \Box E$$

这等于我们可以 no assumption 推出所有 proposition 都是 valid，因此这个 elimination rule 有点太强了. 那么我们考虑让它弱一点，第二次尝试

$$\frac{\Gamma \vdash \Box A\ true}{\Gamma \vdash A\ true}\ \Box E$$

这里确实是 local soundness，但却不是 local completeness

$$\frac{\dfrac{\Gamma \vdash \Box A\ true}{\Gamma \vdash A\ true}\ \Box E}{\Gamma \vdash \Box A\ true}\ ?$$

我们得改变一下思路，如果 $A\ vaild$，那么其他 premise 包含 $A\ vaild$ 的 judgement 那么实际上都是可以去掉 $A\ vaild$，但也仅仅局限以此，这才是 emilination 故事的主线.

Definition 321 Then general judgement form

$$\underbrace{u_1 :: B_1\ vaild, \cdots, u_k :: B_k\ vaild}_{\Delta}; \underbrace{x_1 : A_1\ true, \cdots, x_n : A_n\ true}_{\Gamma} \vdash C\ true$$

Definition 322 The introduction rule and elimination rule of $A\ vaild$ as follow

$$\frac{\Delta; \bullet \vdash A\ true}{\Delta; \Gamma \vdash \Box A\ true}\ \Box I \qquad \frac{\Delta; \Gamma \vdash \Box A\ true \quad \Delta, u :: A\ vaild; \Gamma \vdash C\ true}{\Delta; \Gamma \vdash C\ true}\ \Box E$$

Theorem 323 Local soundness and local completeness of above introduction and elimination rule are held

Annotation 324 可以看到 emilination rule 变成了 substitution，而不是从单纯从本身要得到什么，后面会看见更多这样的东西.

Example 325 Proof of $\cdot; \cdot \vdash \Box A \supset A$.

$$\dfrac{\dfrac{\dfrac{}{\cdot; x : \Box A\ true \vdash \Box A\ true}\ x \qquad \dfrac{}{u :: A\ vaild; x : \Box A\ true \vdash A\ true}\ u}{\cdot; x : \Box A\ true}\ \Box E^u}{\cdot; \cdot \vdash (\Box A \supset A)\ true}\ \supset I^x$$

6.3 Tableaux Calculus

6.4 Possibility

Definition 326 We use $\Diamond A$ for possibility modality.

Annotation 327 $\Diamond A$ 就是一个 claim A is possible 的命题. 通常在 classic modal logic 里面我们定义 A is possible if its negation is not necessary, that is $\Diamond A = \neg\Box\neg A$. 但是这种手法在现在我们讨论的 intuitionistic logic 无法奏效，我们希望的是有一个直观的 introduction rule 来得到它，也就是我们需要一些 explicit evidences，一开始就它的 negation 那显然是做不到的.

Definition 328 [7]The definition of possiblity.

$$\frac{\Delta, \Gamma \vdash A\ true}{\Delta, \Gamma \vdash A\ poss}\ poss$$

Definition 329 The introduction and emilination rule of possiblity.

$$\frac{\Delta; \Gamma \vdash A\ poss}{\Delta; \Gamma \vdash \Diamond A\ true}\ \Diamond I \qquad \frac{\Delta; \Gamma \vdash \Diamond A\ true \quad \Delta; x : A\ true \vdash C\ poss}{\Delta; \Gamma \vdash C\ poss}\ \Diamond E$$

Annotation 330 注意这里的 emilination rule 里面的第二个 premise 中的 hypothesis 只有 $A\ true$，即我们 under assumption $A\ true$, we conclude $C\ poss$.

Theorem 331 Local soundness and completeness are held.

Annotation 332 td; 对上述 inference rule 的理解.

Example 333 Proof of $\Box(A \supset B) \supset \Diamond A \supset \Diamond B$.

参考文献

[1] John Slaney. The Logic Notes.
http://users.cecs.anu.edu.au/~jks/LogicNotes/

[2] The relation between deduction theorem and discharged.
https://math.stackexchange.com/questions/3527285/what-does-discharging-an-assumption-mean-in-natural-deduction

[3] Definition:Discharged Assumption.
https://proofwiki.org/wiki/Definition:Discharged_Assumption

[4] Propositional Logic: Semantics.
https://cs.uwaterloo.ca/~cbruni/CS245Resources/lectures/2018_Fall/05_Propositional_Logic_Semantics_Continued_post.pdf

[5] Propositional Logic: Soundness and Completeness for Natural Deduction.
https://cs.uwaterloo.ca/~cbruni/CS245Resources/lectures/2018_Fall/09_Propositional_Logic_Natural_Deduction_Soundness_and_Completeness_post.pdf

[6] Lecture Notes on Proofs as Programs.
http://www.cs.cmu.edu/~fp/courses/15816-s10/lectures/02-pap.pdf

[7] Computational Interpretations of Modalities.
http://www.cs.cmu.edu/~fp/courses/15816-s10/lectures/04-compmodal.pdf

[8] Classic Modal Logic.
http://www.cs.cmu.edu/~fp/courses/15816-s10/lectures/05-pml.pdf

[9] Verifications and uses.
https://web2.qatar.cmu.edu/cs/15317/lectures/06-verifications.pdf

[10] Sequent Calculus.
https://web2.qatar.cmu.edu/cs/15317/lectures/07-sequents.pdf

[11] Cut and Identity eliminations.
https://web2.qatar.cmu.edu/cs/15317/lectures/08-cutelimination.pdf

[12] Restricted sequent calculus.
https://web2.qatar.cmu.edu/cs/15317/lectures/09-simplified.pdf

[13] Invertibility.
https://web2.qatar.cmu.edu/cs/15317/lectures/10-invertibility.pdf

[14] J. A. Robinson. A Machine-Oriented Logic Based on the Resolution Principle. J. ACM, 12(1):23–41, 1965.
https://web2.qatar.cmu.edu/~greis/robinson-resolution.pdf

[15] Chapter 12, Resolution.
http://logic.stanford.edu/intrologic/notes/chapter_12.html

[16] Chapter 12, Resolution.
https://plato.stanford.edu/entries/proof-theory-development/#SeqCalLatDev