

数理逻辑笔记

陈鸿峥

2020.08*

目录

1	命题逻辑	2
1.1	自然推断	2
1.2	形式语言	4
1.3	语义	5
1.4	规范形式	5
1.5	SAT求解器	6
2	谓词逻辑	6
2.1	形式语言	6
2.2	逻辑证明	7
2.3	语义	8
3	模型检查验证	10
3.1	线性时序逻辑	10
3.2	模型检查	12
3.3	计算树逻辑	13
4	程序验证	14
4.1	证明论	15
5	二元决策图	16
5.1	约简规则	17
5.2	应用规则	18

*Build 20200801

1 命题逻辑

1.1 自然推断

定义 1 (命题(proposition)). 命题或声明式句子是指可判断为真或者假的句子。不可被分解的(*indecomposable*)命题为原子命题。

关于命题公式的定义在这里不再给出, 注意 \rightarrow 是右结合(right-associative)的, 如 $p \rightarrow q \rightarrow r$ 等价于 $p \rightarrow (q \rightarrow r)$ 。

定义 2 (自然推断(deduction)). 假设有一系列前提(*premise*)公式 $\phi_1, \phi_2, \dots, \phi_n$, 及结论 ψ , 那么推断过程可记为

$$\phi_1, \phi_2, \dots, \phi_n \vdash \psi$$

这一表达式称为一个序列(*sequent*), 若一个证明可以被找到则称它是合法的(*valid*)。

注意 \vdash 读作推出(deduce/derive)/逻辑后承, 而 \implies 读作实质蕴涵¹。可以理解为 \vdash 左侧是一些公理(axiom), 右侧是陈述(statement)

推理的基本规则:

- and-introduction ($\wedge i$): 前提与前提为真

$$\frac{\phi \quad \psi}{\phi \wedge \psi} \wedge i$$

- and-elimination ($\wedge e_i$): 前提与中子成分为真

$$\frac{\phi \wedge \psi}{\phi} \wedge e_1 \quad \frac{\phi \wedge \psi}{\psi} \wedge e_2$$

- negation-introduction ($\neg i$)

$$\frac{\phi}{\neg \neg \phi} \neg i$$

- negation-elimination ($\neg e$)

$$\frac{\neg \neg \phi}{\phi} \neg e$$

- implication-elimination $\rightarrow e$

$$\frac{\phi \quad \phi \rightarrow \psi}{\psi} \rightarrow e$$

¹参考以下资料:

- <https://math.stackexchange.com/questions/2903877/to-vs-vdash-in-logic>
- 逻辑学中, 前提为假而命题为真的推论如何解释? - 罗心澄的回答 - 知乎 <https://www.zhihu.com/question/21020308/answer/16917222>

- implies-introduction $\rightarrow i$

$$\frac{\boxed{\begin{array}{c} \phi \\ \vdots \\ \psi \end{array}}}{\phi \rightarrow \psi} \rightarrow i$$

- or-introduction $\vee e$

$$\frac{\phi}{\phi \vee \psi} \vee i_1 \quad \frac{\psi}{\phi \vee \psi} \vee i_2 \quad \frac{\phi \vee \psi \quad \boxed{\begin{array}{c} \phi \\ \vdots \\ \chi \end{array}} \quad \boxed{\begin{array}{c} \phi \\ \vdots \\ \chi \end{array}}}{\chi} \vee e$$

- bottom/not-elimination

$$\frac{\perp}{\phi} \perp e \quad \frac{\phi \quad \neg \phi}{\perp} \neg e$$

- negation

$$\frac{\boxed{\begin{array}{c} \phi \\ \vdots \\ \perp \end{array}}}{\neg \phi} \neg i$$

例 1. 证明 $p \wedge q, r \vdash q \wedge r$ 是合法的。

分析. 推理过程如下

$$\begin{array}{lll} 1 & p \wedge q & \text{premise} \\ 2 & r & \text{premise} \\ 3 & q & \wedge e_2 \quad 1 \\ 4 & q \wedge r & \wedge i \quad 3, 2 \\ & \frac{\frac{p \wedge q}{q} \wedge e_2 \quad r}{q \wedge r} \wedge i \end{array}$$

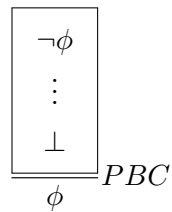
定义 3 (定理(theorem)). 有着合法序列 $\vdash \phi$ 的逻辑公式 ϕ 称为定理。

三条进阶推理规则:

- 拒取式(modus tollens, MT)

$$\frac{\phi \rightarrow \psi \quad \neg \psi}{\neg \phi} MT$$

- 反证法(proof by contradiction, PBC)



- 排中律(the law of the excluded middle, LEM)

$\phi \vee \neg\phi$ 必有一个为真

定义 4 (可证明等价性(provably equivalent)). 令 ϕ 和 ψ 为命题逻辑公式, ϕ 和 ψ 是可证明等价的当且仅当序列 $\phi \vdash \psi$ 和 $\psi \vdash \phi$ 都是合法的, 或者 $\phi \dashv\vdash \psi$

1.2 形式语言

定义 5 (合式公式(well-formed formula, WFF)). 一个WFF是

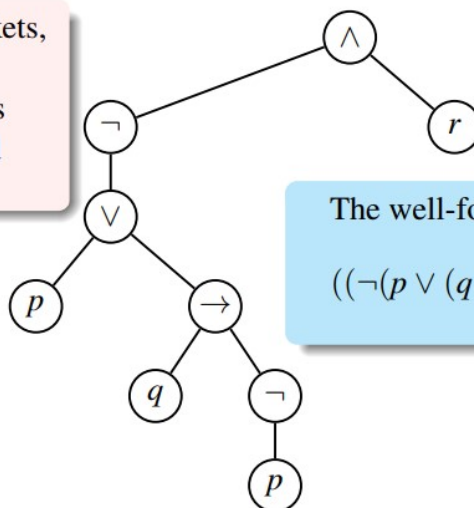
- 一个原子公式 (无论是命题常元还是命题变元)
- 形如 $(\neg\phi)$ 的公式, 其中 ϕ 是一个WFF
- 形如 $(\phi \vee \psi)$ 的公式, 即由二元连接词连接的两个WFF

合式公式可用BNF(Backus Naur Form)定义

$$\phi ::= p \mid \neg\phi \mid \phi \wedge \psi \mid \phi \vee \psi \mid \phi \rightarrow \psi \mid (\phi)$$

The following tree is a parse tree of a well-formed formula.

If we ignore brackets, the in-order representation of this tree is a well-formed formula as a list.



The well-formed formula is

$((\neg(p \vee (q \rightarrow (\neg p)))) \wedge r).$

FIG: A parse tree representing a well-formed formula



1.3 语义

定义 6 (模型(model)). 在前提 $\phi_1, \phi_2, \dots, \phi_n$ 和结论 ψ 上定义另一关系, 记作

$$\phi_1, \phi_2, \dots, \phi_n \models \psi$$

真值包括两个元素 T 和 F , 公式 ϕ 的模型(model)或估值(valuation)是指对 ϕ 中的每一原子命题都有一个真值指派(assignment)。对 $\phi_1, \phi_2, \dots, \phi_n$ 的真值指派决定了 ψ 的真值, 称为 ψ 的解释(interpretation), 可表示为真值表中的一行。

如果对于所有 $\phi_1, \phi_2, \dots, \phi_n$ 的估值都为 T , ψ 也估值为 T , 那么称

$$\phi_1, \phi_2, \dots, \phi_n \models \psi$$

成立(hold), 且称 \models 为语义包含(semantic entailment)关系。

定理 1 (正确性(soundness)). 令 $\phi_1, \phi_2, \dots, \phi_n$ 和 ψ 都是命题逻辑公式, 若 $\phi_1, \phi_2, \dots, \phi_n \vdash \psi$ 是合法的², 那么 $\phi_1, \phi_2, \dots, \phi_n \models \psi$ 成立。(事实上这两者是等价关系)

定义 7 (恒真式(tautology)/矛盾式(contradiction)). 命题逻辑 ϕ 被称为恒真式当且仅当它在所有估值下都取值为 T , 也即 $\models \phi$ 。若在某个估值/解释 I_0 下值为 T , 则称其可满足。若所有估值均为 F , 则为矛盾式。

定理 2. 若 $\models \eta$ 成立, 则 $\vdash \eta$ 是合法的。换句话说, 若 η 是永真式, 则 η 是定理。

1.4 规范形式

定义 8 (语义等价). ϕ 和 ψ 都是命题逻辑的公式, 称其等价当且仅当 $\phi \models \psi$ 和 $\psi \models \phi$ 成立, 记作 $\phi \equiv \psi$, 也等价于 $\vdash (\phi \rightarrow \psi) \wedge (\psi \rightarrow \phi)$ 成立。

定义 9 (合取范式(conjunction normal form, CNF)). BNF定义如下:

- 文字(literal): $L ::= p \mid \neg p$
- 句子(clause): $D ::= L \mid L \vee D$
- 公式(formula): $C ::= D \mid (D) \mid D \wedge C$

例子如

$$(p \vee r) \wedge (\neg p \vee r) \wedge (p \vee \neg r)$$

定义 10 (霍尔公式(Horn formula)). 若命题逻辑公式 ϕ 能用下面的语法, 表示成 H 的一个示例

$$P ::= \perp \mid \top \mid p \quad A ::= P \mid P \wedge AC ::= A \rightarrow PH ::= C \mid C \wedge H$$

则称 C 的每个实例为霍尔从句(clause)。

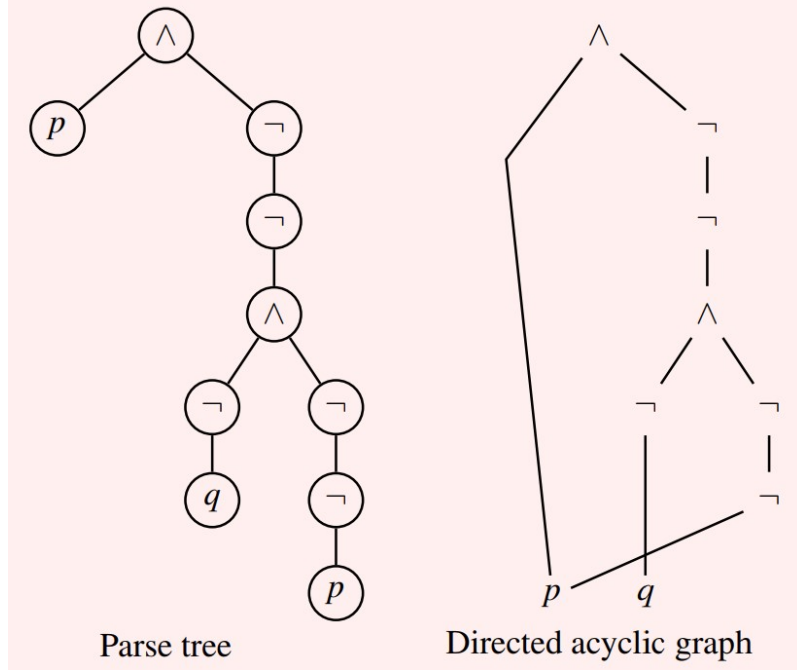
²假设 ϕ_1, \dots, ϕ_n 为真, 可以推出 ψ 为真

1.5 SAT求解器

线性求解器只接受以下几种形式的公式

$$\phi ::= p \mid (\neg\phi) \mid (\phi \wedge \phi)$$

例 2. $\phi = p \wedge \neg(q \vee \neg p)$, 计算 $T(\phi) = p \wedge \neg\neg(\neg q \wedge \neg\neg p)$, 则有语法树和DAG如下



2 谓词逻辑

谓词逻辑(predicate)或被称为一阶逻辑, 提供了更强的语言表达能力。

2.1 形式语言

定义 11 (项(item)). 项的定义如下:

- 每一个变量都是项
- 若 $c \in \mathcal{F}$ 是一个空函数, 则 c 是项 (常数)
- 若 t_1, t_2, \dots, t_n 都是项, 且 $f \in \mathcal{F}$ 有 $n > 0$ 个元 (arity), 则 $f(t_1, t_2, \dots, t_n)$ 是一个项

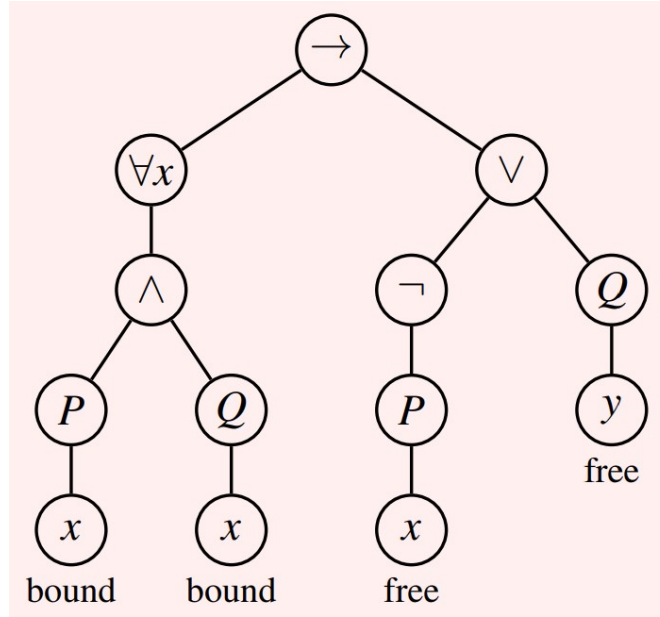
用BNF写即

$$t ::= x \mid c \mid f(t_1, t_2, \dots, t_n)$$

定义 12 (公式(formula)). BNF定义如下

$$\phi ::= P(t_1, t_2, \dots, t_n) \mid (\neg\phi) \mid (\phi \wedge \psi) \mid (\phi \vee \psi) \mid (\phi \rightarrow \psi) \mid (\forall x\phi) \mid (\exists x\phi)$$

定义 13 (自由(free)/约束(bound)变量). 语法树叶结点往上不会经过 $\forall x$ 或 $\exists x$ 结点, 则为自由变量。



定义 14 (替代(substituion)). 给定变量 x 、项 t 和公式 ϕ , 定义 $\phi[t/x]$ 为将 ϕ 中所有自由 x 用 t 代替

2.2 逻辑证明

- equality ($= i$)

$$\frac{}{t = t} = i$$

- substitution ($= e$)

$$\frac{t_1 = t_2 \quad \phi[t_1/x]}{\phi[t_2/x]} = e$$

- for-eliminating ($\forall x \quad e$)

$$\frac{\forall x \phi}{\phi[t/x]} \forall x \quad e$$

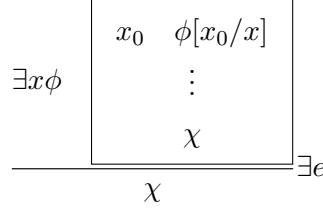
- for-introduction ($\forall x \quad i$)

$$\boxed{\begin{array}{c} x_0(\text{fresh/dummy var}) \\ \vdots \\ \phi[x_0/x] \end{array}} \forall x \quad i$$

- exists-introduction ($\exists x \quad i$)

$$\frac{\phi[t/x]}{\exists x \phi} \exists x \quad i$$

- exists-elimination ($\exists e$)



例 3. 证明 $\forall x(P(x) \rightarrow Q(x)), \forall xP(x) \vdash \forall xQ(x)$

分析. 推理过程如下

1	$\forall x(P(x) \rightarrow Q(x))$	premise
2	$\forall xP(x)$	premise
3	$P(x_0) \rightarrow Q(x_0)$	$\forall x \quad e \quad 1$
4	$P(x_0)$	$\forall x \quad e \quad 2$
5	$Q(x_0)$	$\rightarrow e \quad 3, 4$
6	$\forall xQ(x)$	$\forall x \quad i \quad 3-5$

定理 3 (等价性(equivalence)). 令 ϕ 和 ψ 都为谓词逻辑的公式, 有以下等价性

$$\neg \forall x\phi \vdash \exists x\neg\phi, \quad \neg \exists x\phi \vdash \forall x\neg\phi$$

2.3 语义

记 Γ 为公式 ϕ_1, \dots, ϕ_n , 要证明 $\Gamma \vdash \psi$ 是合法的, 则只需从 Γ 中提供 ψ 的证明。而如果要证明 Γ 推不出 ψ , 从证明论(proof theory)的角度是困难的。

但从语义(semantics)的角度, 只需找到一个模型(model)满足所有 ϕ_i 都为真, 而 ψ 为假, 即可得到 Γ 推不出 ψ ; 相反地, 要证明 Γ 推出 ψ 则是困难的, 因为对于谓词逻辑来说有无穷多种估值/模型, 只有全部验证了才能得知 $\Gamma \models \psi$ (ψ 被 Γ 语义蕴含 entail)。

因此证明论和模型论两者都是重要的。

定义 15 (模型(model)). 令 \mathcal{F} 为函数符号的集合, \mathcal{P} 为谓词符号的集合³。关于 $(\mathcal{F}, \mathcal{P})$ 的模型 \mathcal{M} 包含以下数据:

- 非空集合 A : 具体值的全集
- 对于每一空函数符号 $f \in \mathcal{F}$, $f^{\mathcal{M}} \in A$
- 对于每一 $f \in \mathcal{F}$ 且元 $n > 0$, 具体函数 $f^{\mathcal{M}} : A^n \rightarrow A$
- 对于每一 $P \in \mathcal{P}$ 且元 $n > 0$, 子集 $P^{\mathcal{M}} \subset A^n$

也就是说 f 和 P 仅仅是抽象的符号, 而 $f^{\mathcal{M}}$ 和 $P^{\mathcal{M}}$ 为具体的函数/元素。也可以说模型给出了一个解释(interpretation)。

³函数符号是一个算子, 作用在项上并生成一个新的项/实体(object), 比如+和 \times ; 而谓词符号也是一个算子, 作用在项上并生成一个谓词/宣称(claim), 比如<和>。

例 4. 令 $\mathcal{F} \stackrel{\text{def}}{=} \{i\}$, $\mathcal{P} \stackrel{\text{def}}{=} \{R, F\}$, 其中 i 是一个常数, F 是一元谓词符号, R 是二元谓词符号。一个模型可以是

$$\begin{aligned} A &\stackrel{\text{def}}{=} \{a, b, c\} \\ i^{\mathcal{M}} &\stackrel{\text{def}}{=} a \\ R^{\mathcal{M}} &\stackrel{\text{def}}{=} \{(a, a), (a, b), (a, c), (b, c), (c, c)\} \\ F^{\mathcal{M}} &\stackrel{\text{def}}{=} \{b, c\} \end{aligned}$$

那么有 $\exists y R(i, y)$ 为真, $\neg F(i)$ 为真。

上面给出了模型的定义, 并且使得我们可以直接从 $(\mathcal{F}, \mathcal{P})$ 中计算出真值, 但我们仍需讨论如何处理全称量词 $\forall x \phi$ 及特称量词 $\exists x \phi$, 需要检查 ϕ 是否对于所有模型中的 a 都成立。尽管我们可以用 $\phi[a/x]$ 表示, 但是 $\phi[a/x]$ 并不是一个逻辑公式, 因为 a 不是一个项(term)而是模型中的一个元素。

因此需要限定公式是关于一个环境的(relative to an environment)。

定义 16 (环境(environment)/查找表(look-up table)). 对于全集 A 具体值的环境是一个函数 $l : \text{var} \rightarrow A$, 定义 $l[x \mapsto a]$ 为从 x 映射到 a 的查找表。

定义 17 (满足关系(satisfaction)). 给定 $\mathcal{M}(\mathcal{F}, \mathcal{P})$ 及环境 l , 定义满足关系 $\mathcal{M} \models_l \phi$ 。若 $\mathcal{M} \models_l$ 成立(hold), 则称 ϕ 在关于环境 l 的模型 \mathcal{M} 中计算为 T 。

例 5. 令 $F \stackrel{\text{def}}{=} \{alma\}$, $P \stackrel{\text{def}}{=} \{\text{loves}\}$, 其中 $alma$ 是常数, $\text{loves}(\cdot, \cdot)$ 是谓词符号。模型 \mathcal{M} 包含集合

$$A \stackrel{\text{def}}{=} \{a, b, c\}, alma^{\mathcal{M}} \stackrel{\text{def}}{=} a, \text{loves}^{\mathcal{M}} \stackrel{\text{def}}{=} \{(a, a), (b, a), (c, a)\}$$

检查模型 \mathcal{M} 是否满足

$$\text{None of Alma's lovers' lovers love her.}$$

$$\phi : \forall x \forall y (\text{loves}(x, alma) \wedge \text{loves}(y, x) \rightarrow \neg \text{loves}(y, alma))$$

令 $x \mapsto a$ 及 $y \mapsto b$, 又 $alma^{\mathcal{M}} \stackrel{\text{def}}{=} a$, 得到

$$\text{loves}(a, a) \wedge \text{loves}(b, a) \rightarrow \neg \text{loves}(b, alma)$$

不成立, 故 $\mathcal{M} \not\models \phi$

在命题逻辑中, $\phi_1, \dots, \phi_n \models \psi$ 当且仅当 ϕ_1, \dots, ϕ_n 都估值为 T , 同时 ψ 也估值为 T 。

定义 18. 令 Γ 为谓词逻辑的公式集合, ψ 为谓词逻辑的公式

1. 语义蕴含 $\Gamma \models \psi$ 成立当且仅当对于所有的模型 \mathcal{M} 和查找表 l , 只要 $\forall \phi \in \Gamma : \mathcal{M} \models_l \phi$ 成立, $\mathcal{M} \models_l \psi$ 也成立
2. ψ 是可满足的(satisfiable)当且仅当存在模型 \mathcal{M} 和环境 l 使得 $\mathcal{M} \models_l \psi$ 成立
3. ψ 是合法的(valid)当且仅当 $\mathcal{M} \models_l \psi$ 对于所有模型 \mathcal{M} 和环境 l 都成立

⁴ \models 代表语义蕴含(semantic entailment)

例 6. 考虑以下语义蕴含关系

$$\forall x(P(x) \rightarrow Q(x)) \models \forall xP(x) \rightarrow \forall xQ(x)$$

令 \mathcal{M} 为满足 $\forall x(P(x) \rightarrow Q(x))$ 的模型, 需要证明 \mathcal{M} 也满足 $\forall xP(x) \rightarrow \forall xQ(x)$

- 若不是每个 \mathcal{M} 中的元素都满足 P , 则前件为假, 显然成立
- 若每个 \mathcal{M} 中的元素都满足 P , 则每一元素都满足 Q , 因为 \mathcal{M} 满足 $\forall x(P(x) \rightarrow Q(x))$

进而 $\mathcal{M} \models \forall xP(x) \rightarrow \forall xQ(x)$

例 7. 考虑以下语义蕴含关系

$$\forall xP(x) \rightarrow \forall xQ(x) \models \forall x(P(x) \rightarrow Q(x))$$

令 $A \stackrel{\text{def}}{=} \{a, b\}, P^{\mathcal{M}} \stackrel{\text{def}}{=} \{a\}, Q^{\mathcal{M}} \stackrel{\text{def}}{=} \{b\}$, 则 $\mathcal{M} \models \forall xP(x) \rightarrow \forall xQ(x)$, 但 $\mathcal{M} \not\models \forall x(P(x) \rightarrow Q(x))$

3 模型检查验证

3.1 线性时序逻辑

定义 19 (线性时序逻辑(linear-time temporal logic, LTL)). BNF定义如下

$$\phi ::= \top \mid \perp \mid p \mid (\neg\phi) \mid (\phi \wedge \phi) \mid (\phi \vee \phi) \mid (\phi \rightarrow \phi) \mid (X\phi) \mid (F\phi) \mid (G\phi) \mid (\phi U \phi) \mid (\phi W \phi) \mid (\phi R \phi)$$

其中 X, F, G, U, R, W 都称为时序连接词(*temporal connectives*)

- X : *next state*
- F : *some Future state* (存在)
- G : *all future states (Globally)* (任意)
- U : *Until* (二元)
- R : *Release* (二元)
- W : *Weak-until* (二元)

运算符优先级

- 一元连接词: \neg, X, F, G
- U, R, W
- \wedge, \vee
- \rightarrow

定义 20 (转移系统(transition system)). 一个转移系统 $\mathcal{M} = (S, \rightarrow, L)$ 是状态集合 S (静态结构), 转移关系 \rightarrow (动态结构), 使得 $\forall s \in S, \exists s' \in S: s \rightarrow s'$, 且有标注函数 $L: S \rightarrow \mathcal{P}(Atoms)$, 其中 $\mathcal{P}(Atoms)$ 为原子描述的幂集 (实际上 L 就是给所有命题原子做真值指派)。转移系统也可以被称作模型(*model*)。

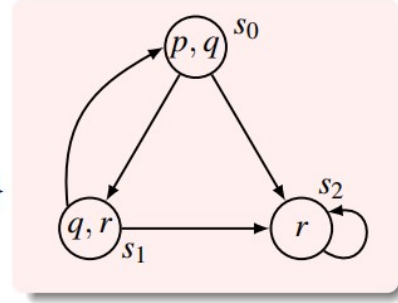
(*) $\mathcal{P}(\mathbf{Atoms})$ is the power set of \mathbf{Atoms} .

Let $\mathcal{M} = (S, \rightarrow, L)$, where

$$S = \{s_0, s_1, s_2\}$$

$$\rightarrow = \{(s_0, s_1), (s_0, s_2), (s_1, s_0), (s_1, s_2), (s_2, s_2)\}$$

$$L = \{(s_0, \{p, q\}), (s_1, \{q, r\}), (s_2, \{r\})\}$$



定义 21 (路径(path)). 路径是一个无穷序列 $s_1, s_2, \dots \in S$ 使得 $s_i \rightarrow s_{i+1}, \forall i \geq 1$ 。令 $\pi^i = s_i \rightarrow s_{i+1} \rightarrow \dots$ 为从状态 s_i 开始的路径。路径 $\pi = s_1 \rightarrow s_2 \rightarrow \dots$ 满足 LTL 公式定义在满足关系 \models 上:

1. $\pi \models \top$
2. $\pi \not\models \perp$
3. $\pi \models p$ iff $p \in L(s_1)$
4. $\pi \models \neg\phi$ iff $\pi \not\models \phi$
5. $\pi \models \phi \wedge \psi$ iff $\pi \models \phi$ and $\pi \models \psi$
6. $\pi \models \phi \vee \psi$ iff $\pi \models \phi$ or $\pi \models \psi$
7. $\pi \models \phi \rightarrow \psi$ iff $\pi \models \psi$ when $\pi \models \phi$
8. $\pi \models X\phi$ iff $\pi^2 \models \phi$
9. $\pi \models G\phi$ iff $\forall i \geq 1: \pi^i \models \phi$
10. $\pi \models F\phi$ iff $\exists i \geq 1: \pi^i \models \phi$
11. $\pi \models \phi U \psi$ iff $\exists i \geq 1: \pi^i \models \psi$ and $\forall j = 1, \dots, i-1: \pi^j \models \phi$
12. $\pi \models \phi W \psi$ iff **either** $\exists i \geq 1: \pi^i \models \psi$ and $\forall j = 1, \dots, i-1: \pi^j \models \phi$ **or** $\forall k \geq 1: \pi^k \models \phi$
13. $\pi \models \phi R \psi$ iff **either** $\exists i \geq 1: \pi^i \models \phi$ and $\forall j = 1, \dots, i: \pi^j \models \psi$ **or** $\forall k \geq 1: \pi^k \models \psi$

记 $\mathcal{M}, s \models \phi$ 表示对于所有 \mathcal{M} 开始于 s 的执行路径 π , 都有 $\pi \models \phi$, 或可简写为 $s \models \phi$

注意有以下恒等式

$$\neg G\phi \equiv F\neg\phi$$

$$\neg F\phi \equiv G\neg\phi$$

$$\neg X\phi \equiv X\neg\phi$$

$$\phi W\psi \equiv \phi U\psi \vee G\phi$$

$$\phi R\psi \equiv \neg(\neg\phi U\neg\psi)$$

$$F(\phi \vee \psi) \equiv F\phi \vee F\psi$$

$$G(\phi \wedge \psi) \equiv G\phi \wedge G\psi$$

$$F\phi \equiv \top U\phi$$

$$G\phi \equiv \perp R\phi$$

$$\phi U\psi \equiv \phi W\psi \wedge F\psi$$

$$\phi W\psi \equiv \phi U\psi \vee G\phi$$

$$\phi W\psi \equiv \psi R(\phi \vee \psi)$$

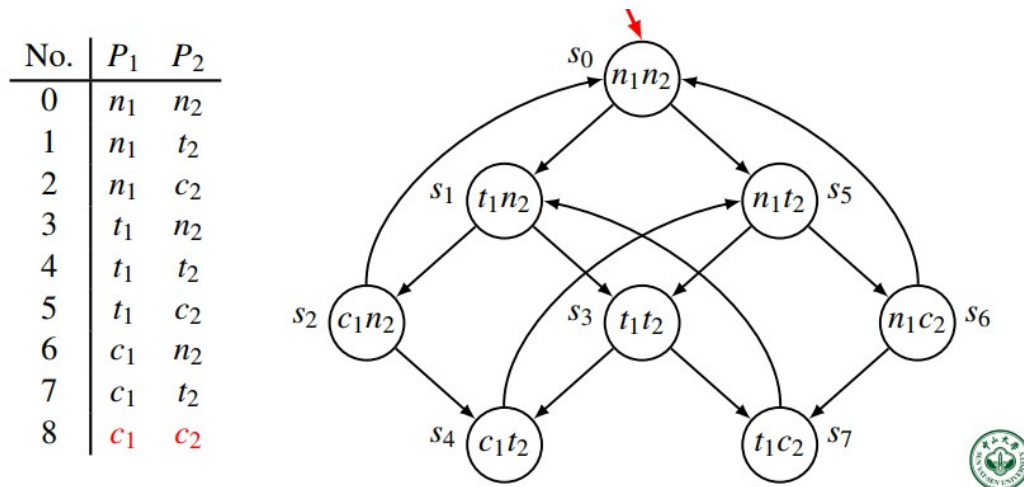
$$\phi R\psi \equiv \psi W(\phi \wedge \psi)$$

例 8. 对于上图的例子, 有以下式子成立

- $s_0 \models p \wedge q$ (3,5)
- $s_0 \models Xr$, 因 $r \in L(s_1), L(s_2)$
- $s_0 \models G\neg(p \wedge r)$, 因所有从 s_0 开始的路径都满足 $G\neg(p \wedge r)$, 也即满足 $\neg(p \wedge r)$

3.2 模型检查

下面是一个互斥锁的例子, n 为非临界区状态, t 为尝试进入临界区, c 为临界区状态。考虑2个进程, 每个进程的转移都是 $n \rightarrow t \rightarrow c \rightarrow n \rightarrow \dots$ 。



有以下性质:

- 安全性(safety): $G\neg(c_1 \wedge c_2)$ 在每个状态都被满足

- 活性(liveness): $G(t_1 \rightarrow Fc_1)$, 对于路径 $s_0 \rightarrow s_1 \rightarrow s_3 \rightarrow s_7 \rightarrow s_1 \rightarrow s_3 \rightarrow \dots$ 就不满足
- 非阻塞(non-blocking): 对于任意状态满足 n_1 , 有后继状态满足 t_1 , 无法用LTL表达
- 非严格序列(strict sequencing)

3.3 计算树逻辑

定义 22 (计算树逻辑(computation tree logic, CTL)). 除了LTL有的 U, F, G, X , CTL还有 A 和 E 表示所有路径(*All path*)和存在一条路径(*Exist a path*)。BNF定义如下

$$\begin{aligned} \phi ::= & \top \mid \perp \mid p \mid (\neg\phi) \mid (\phi \wedge \phi) \mid (\phi \vee \phi) \mid (\phi \rightarrow \phi) \mid (AX\phi) \mid (EX\phi) \mid \\ & (AF\phi) \mid (EF\phi) \mid (AG\phi) \mid (EG\phi) \mid A[\phi U \phi] \mid E[\phi U \phi] \end{aligned}$$

运算符优先级

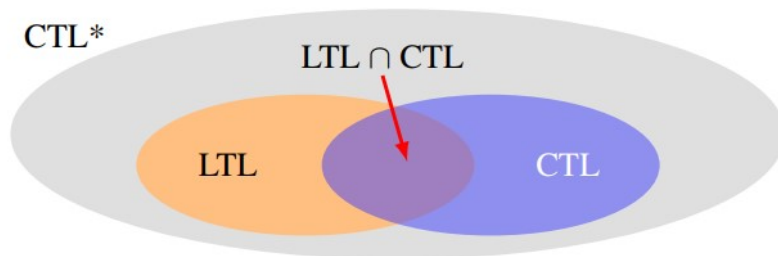
- $\neg, AG, EG, AF, EF, AX, EX$
- \wedge, \vee
- \rightarrow, AU, EU

定义 23 (CTL*). ϕ 为状态公式(在状态上求值), α 为路径公式(在路径上求值)

$$\begin{aligned} \phi ::= & \top \mid p \mid (\neg\phi) \mid (\phi \wedge \phi) \mid A[\alpha] \mid E[\alpha] \\ \alpha ::= & \phi \mid (\neg\alpha) \mid (\alpha \wedge \alpha) \mid (\alpha U \alpha) \mid (G\alpha) \mid (F\alpha) \mid (X\alpha) \end{aligned}$$

- LTL is a subset of CTL*
- The LTL formula α is equivalent to the CTL* formula $A[\alpha]$.
- CTL is a subset of CTL*
- The CTL formula is the fragment of CTL* in which we restrict the form of path formulas.

$$\alpha ::= (\alpha U \alpha) \mid (G\alpha) \mid (F\alpha) \mid (X\alpha)$$



The expressive powers of LTL, CTL and CTL*



4 程序验证

定义 24 (霍尔三元组(Hoare triple)).

$$\langle \phi \rangle P \langle \psi \rangle$$

代表若程序 P 在满足 ϕ 的状态下运行, 则执行完 P 的状态会满足 ψ 。 ϕ 称为先验条件(*precondition*), ψ 称为后验条件(*postcondition*)。存储变量 x 记为 $l(x)$ 。

例 9. 若输入 x 是正数, 则求出一个数字它的平方小于 x 。

记程序为 P , 则Hoare三元组为

$$\langle x > 0 \rangle P \langle y \cdot y < x \rangle$$

一个可行的程序 P 可以是

```
y = 0;
while (y * y < x) { y = y + 1; }
y = y - 1;
```

定义 25 (部分正确性(*partial correctness*)). 若对于所有满足先验条件 ϕ 的状态, 只要 P 需要能停止(*terminate*), 经过 P 的执行, 都满足后验条件 ψ , 则 $\langle \phi \rangle P \langle \psi \rangle$ 满足部分正确性。若 $\models_{par} \langle \phi \rangle P \langle \psi \rangle$ 成立, 则称 \models_{par} 为部分正确性关系。全部正确性(*total correctness*)则是保证了 P 一定会停止。

例 10. 考虑下面求阶乘的程序 $Fac1$:

```
y = 1;
z = 0;
while (z != x) {
    z = z + 1;
    y = y * z;
}
```

- $\models_{tot} \langle x \geq 0 \rangle Fac1 \langle y = x! \rangle$ 成立, 只要 $x \geq 0$, $Fac1$ 一定会停止, 并且有结果 $y = x!$
- $\models_{tot} \langle \top \rangle Fac1 \langle y = x! \rangle$ 不保证成立, 因为 $Fac1$ 对于 x 的负数值不会停止
- $\models_{par} \langle x \geq 0 \rangle Fac1 \langle y = x! \rangle$ 和 $\models_{par} \langle \top \rangle Fac1 \langle y = x! \rangle$ 成立

定义 26 (逻辑变量(*logical variable*)). 逻辑变量在 ϕ 或 ψ 是自由的, 且不出现在 P 中

例 11. $\langle x = x_0 \wedge x \geq 0 \rangle Sum \langle z = x_0(x_0 + 1)/2 \rangle$

```
z = 0;
while (x != 0) {
    z = z + x;
    x = x - 1;
}
```

4.1 证明论

- Composition

$$\frac{\langle\phi\rangle C_1 \langle\eta\rangle \quad \langle\eta\rangle C_2 \langle\psi\rangle}{\langle\phi\rangle C_1; C_2 \langle\psi\rangle}$$

- Assignment

$$\overline{\langle\psi[E/x]\rangle x = E \langle\psi\rangle}$$

注意这条公式相当tricky，是在先验条件中将 x 换为 E （即恢复 $x = E$ 的赋值），在具体证明中往往是反过来用

- If-statement

$$\frac{\langle\phi \wedge B\rangle C_1 \langle\psi\rangle \quad \langle\phi \wedge \neg B\rangle C_2 \langle\psi\rangle}{\langle\phi\rangle \text{ if } B\{C_1\} \text{ else } \{C_2\} \langle\psi\rangle}$$

- Partial-while

$$\frac{\langle\psi \wedge B\rangle C \langle\psi\rangle}{\langle\psi\rangle \text{ while } B\{C\} \langle\psi \wedge \neg B\rangle}$$

- Implied

$$\frac{\vdash_{AR} \phi' \rightarrow \phi \quad \langle\phi\rangle C \langle\psi\rangle \quad \vdash_{AR} \psi \rightarrow \psi'}{\langle\phi'\rangle C \langle\psi'\rangle}$$

基本谓词逻辑演算及算术表达都满足

例 12 (表格证明(proof tableaux)). $\vdash_{par} \langle y < 3 \rangle y = y + 1 \langle y < 4 \rangle$

$$\begin{array}{l} \langle y < 3 \rangle \\ \langle y + 1 < 4 \rangle \quad \text{Implied} \\ y = y + 1 \\ \langle y < 4 \rangle \quad \text{Assignment} \end{array}$$

$$\langle\phi\rangle \text{ if } (B)\{C_1\} \text{ else } \{C_2\} \langle\psi\rangle$$

- 将 ψ 反推 C_1 ，结果记为 ϕ_1
- 将 ψ 反推 C_2 ，结果记为 ϕ_2
- 令 $\phi = (B \rightarrow \phi_1) \wedge (\neg B \rightarrow \phi_2)$

例 13. 考虑程序 $Succ$

```
a = x + 1;
if (a-1 == 0) { y = 1; } else { y = a; }
```

证明 $\vdash_{par} \langle \top \rangle Succ \langle y = x + 1 \rangle$ 合法

分析. 可以自底向上分析

$\langle \top \rangle$

```

 $\langle\langle x + 1 - 1 = 0 \rightarrow 1 = x + 1 \rangle \wedge (\neg(x + 1 - 1 = 0) \rightarrow x + 1 = x + 1)\rangle$  Implied
a = x + 1;
 $\langle\langle a - 1 = 0 \rightarrow 1 = x + 1 \rangle \wedge (\neg(a - 1 = 0) \rightarrow a = x + 1)\rangle$  Assignment
if (a - 1 == 0) {
     $\langle 1 = x + 1 \rangle$  If-statements
    y = 1;
     $\langle y = x + 1 \rangle$  Assignment
} else {
     $\langle a = x + 1 \rangle$  If-statements
    y = a;
     $\langle y = x + 1 \rangle$  Assignment
}
 $\langle y = x + 1 \rangle$  If-statements

```

$\langle\phi\rangle$ while $B\{C\}$ $\langle\psi\rangle$

需要找到合适的不变量(invariant) η 使得

- $\models_{AR} \phi \rightarrow \eta$
- $\models_{AR} \eta \wedge \neg B \rightarrow \psi$
- $\models_{par} \langle\eta\rangle$ while $(B)\{C\}\langle\eta \wedge \neg B\rangle$

例 14. 考虑程序 *Fac1*

```

y = 1;
z = 0;
while (z != x) { // L1
    z = z + 1;
    y = y * z;
} // L2

```

证明

$$\models_{AR} (y = 1 \wedge z = 0) \rightarrow (y = z!) \quad \models_{AR} (y = z! \wedge x = z) \rightarrow (y = x!)$$

```

 $\langle\top\rangle$ 
 $\langle 1 = 0! \rangle$  Implied
y = 1;
 $\langle y = 0! \rangle$  Assignment
z = 0;
 $\langle y = z! \rangle$  Assignment
while (z != x) {
     $\langle y = z! \wedge z \neq x \rangle$  Invariant Hyp.  $\wedge$  guard
     $\langle y \cdot (z + 1) = (z + 1)! \rangle$  Implied
    z = z + 1;
}

```



```

    ( $y \cdot z = z!$ ) Assignment
   $y = y * z;$ 
    ( $y = z!$ ) Assignment
}
( $y = z! \wedge \neg(z \neq x)$ ) Partial-while
( $y = x!$ ) Implied

```

5 二元决策图

·为与，+为或， \oplus 为异或。

布尔函数用真值表表示，如 $f(x, y) \stackrel{\text{def}}{=} \overline{x + y}$ 。

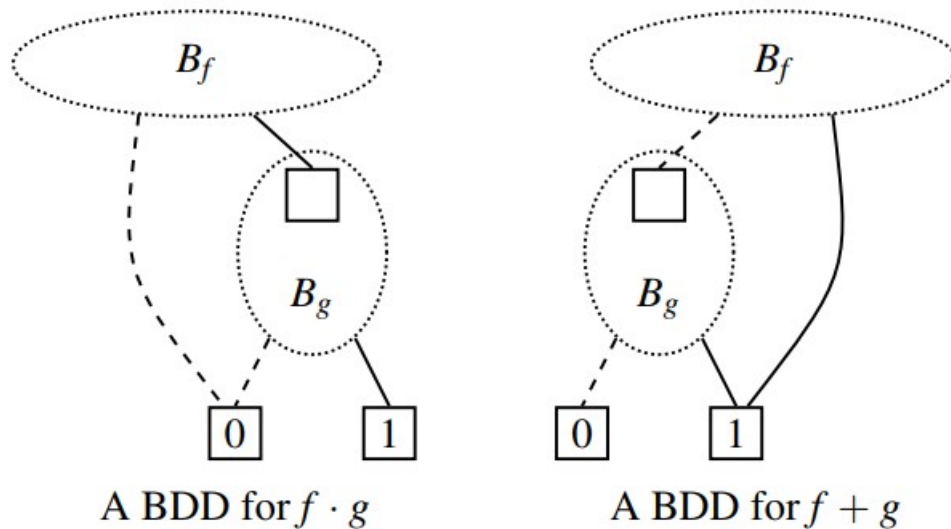
x	y	$f(x, y)$
1	1	0
1	0	1
0	1	0
0	0	1

二元决策图(Binary Decision Diagram, BDD)

- 非终端结点标号为布尔变量
- 终端结点（叶子）标号为0或1
- 每一个非终端结点有两条边，一条虚边(dashed)指向0，一条实边(solid)指向1

简化法则：

- 去除重复终端(terminal)
- 去除冗余测试：若结点 n 出边都指向同个结点 m ，则消除 n
- 去除重复非终端符号：第一条是本条的特例



定义 27 (有序BDD). 令 $[x_1, \dots, x_n]$ 为有序 n 个变量, B 为含有这些变量的BDD

- 令 $V(B)$ 为OBDD的变量集, 即 $V(B) = \{x_1, x_2, \dots, x_n\}$
- $O_B(x_i)$ 是 $\forall x_i \in V(B)$ 的序
- $\forall x_i, x_j \in V(B), O_B(x_i) < O_B(x_j)$ 为在 B 中任一路径每一个 x_i 跟随 x_j

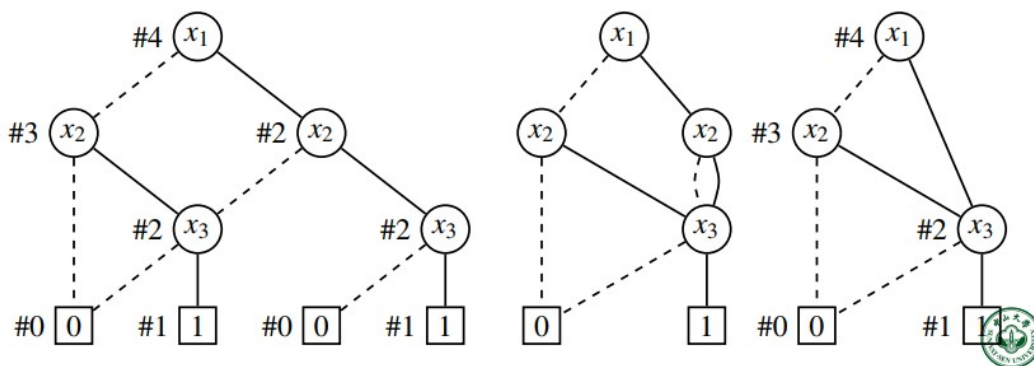
5.1 约简规则

定义 28. 给定BDD中非终端结点 n

- $lo(n)$ 是从 n 用虚线指向的点
- $hi(n)$ 是从 n 用实线指向的点

$id(\cdot)$ 为结点编号

- If $id(lo(n)) = id(hi(n))$, we set $id(n)$ to be that label. That is because the boolean function represented at n is the same function as the one represented at $lo(n)$ and $hi(n)$.
- If $id(lo(n)) = id(lo(m))$ and $id(hi(n)) = id(hi(m))$, where n and m have the same variable x_i , we set $id(n)$ to be $id(m)$.
- Set $id(n)$ to the next unused integer label.



5.2 应用规则

定义 29 (香农展开(Shannon expansion)). 对于所有布尔公式 f 及布尔变量 x (包括那些不出现在 f 中的), 有

$$f \equiv \bar{x} \cdot f[0/x] + x \cdot f[1/x]$$

应用(apply)函数基于 $f \text{ op } g$ 的展开

$$\begin{aligned} f \text{ op } g &\equiv (\bar{x} \cdot f[0/x] + x \cdot f[1/x]) \text{ op } (\bar{x} \cdot g[0/x] + x \cdot g[1/x]) \\ &\equiv (\bar{x} \cdot f[0/x] \text{ op } \bar{x} \cdot g[0/x]) + (x \cdot f[1/x] \text{ op } x \cdot g[1/x]) \\ &\equiv \bar{x} \cdot (f[0/x] \text{ op } g[0/x]) + x \cdot (f[1/x] \text{ op } g[1/x]) \end{aligned}$$

- 特称规则:

$$\exists x.f \stackrel{\text{def}}{=} f[0/x] + f[1/x]$$

- 全称规则:

$$\exists x.f \stackrel{\text{def}}{=} f[0/x] \cdot f[1/x]$$

f	OBDD B_f	f	OBDD B_f
0	B_0	1	B_1
x	B_x	\bar{f}	交换 B_f 中的0和1结点
$f + g$	$apply(+, B_f, B_g)$	$f \cdot g$	$apply(\cdot, B_f, B_g)$
$f \oplus g$	$apply(\oplus, B_f, B_g)$		
$f[0/x]$	$restrict(0, x, B_f)$	$f[1/x]$	$restrict(1, x, B_f)$
$\exists x.f$	$apply(+, B_{f[0/x]}, B_{f[1/x]})$	$\forall x.f$	$apply(\cdot, B_{f[0/x]}, B_{f[1/x]})$