互模拟, (余) 归纳及其他

哲学数学计算机中的逻辑课程 (2016 年秋)

王彦晶

北大哲学系

2016 年 12 月 8 日

互模拟 (bisimulation)

(余) 归纳及其他

框架及其性质

我们说公理 φ 对应于框架性质 X, 如果对任意框架 $\mathcal{F}$:

φ 在框架 $\mathcal{F}$ 上有效 $\iff$ $\mathcal{F}$ 有性质 X:

- T $\Box p \to p$ 自反性 (Reflexivity): 任意 x, xRx
- B $p \to \Box\Diamond p$ 对称性 (Symmetry): 任意 x, y, 若 xRy 则 yRx
- D $\Box p \to \Diamond p$ 持续性 (Seriality): 任意 x, 存在 y 使得 xRy
- 4 $\Box p \to \Box\Box p$ 传递性 (Transitivity) 任意 xyz 若 $xRy\&yRz$ 则 xRz
- 5 $\neg\Box p \to \Box\neg\Box p$ 欧几里得性 (Euclidean property) 任意 xyz 若 xRy 且 xRz 则 yRz　来源于《几何原本》: *things which equal the same thing also equal one another.*

有没有直观的性质不能被模态定义? 当然有! 比如反对称 (anti-symmetry): 对任意 x,y: 如果 xRy 且 yRx 则有 $x = y$.

怎么说一个性质<u>不能</u> 被任何模态公式定义? 又是否定性结果!

要看模态逻辑最多能说啥

一个语言能够说多么细致的事情可以由它区分模型的能力体现.

说 φ 可以区分 两个点模型 $\mathcal{M}, w$ 和 $\mathcal{N}, v$, 如果 φ 在一个上真另一个为假.

说一个逻辑语言 L 可以区分两个点模型, 如果这个语言里存在 一个公式可以区分这两个点模型. 如果不能区分我们就说这两个模型是相对于语言 L逻辑等价 的 $(\mathcal{M}, s \equiv_L \mathcal{N}, v)$

每个逻辑语言根据其语义肯定对应于一种对模型的区分能力.

我们能不能给这个区分关系一个**纯结构** (不涉及逻辑语言) 的刻画? 能否找到这样的的等价关系 $\approx$ 使得 $\equiv_L = \approx$? (L 是模态逻辑的时候) "结构长得像" 和 "说不出来有什么区别" 一样.

可以! 我们会从不同的角度理解这个等价关系!

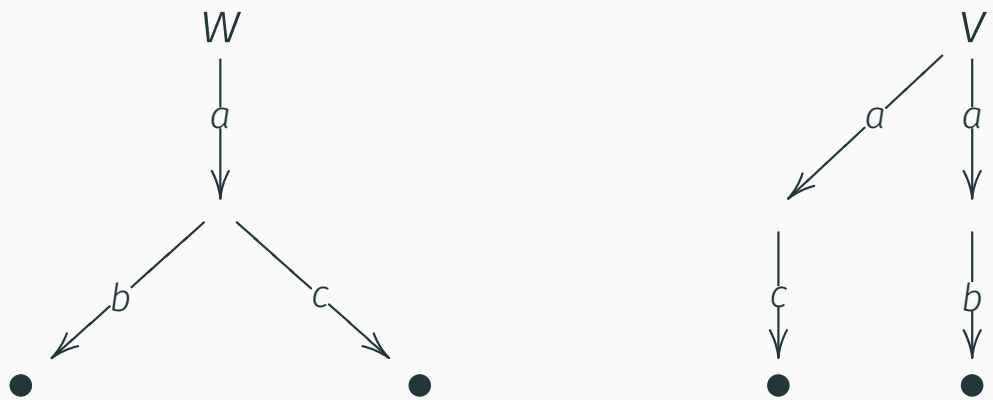
例子

考虑有多个模态词的简单语言

$$\varphi ::= \top \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid [a]\varphi$$

其中 $a \in \Sigma$.

能区分吗?



能! $[a](\langle b\rangle\top \wedge \langle c\rangle\top)$ 在左边模型的 w 上是真的, 但在右边的 v 上是假的, 虽然最后可能的执行路径都是 ab, ac, 但是可选择的点不一样, 应该是不同的程序!

不能区分的

不能区分:

基本的模态逻辑不能数数...

察其言, 观其行而已...

互模拟 (BISIMULATION)

互模拟在三个不同领域的发现

- 模态逻辑: van Benthem (1976) (基于 Segerberg (1971), de Jongh and Troelstra (1966)+ 函数改关系)
- 计算机: Park (1981) (基于 Milner (1980) 的工作 + 不动点)
- 非良基集合论 (non-well-founded) Set theory: Forti and Honsell (1981) Hinnion, Aczel (1988)

非良基集合论与互模拟

什么时候两个集合相等? 他们有同样的元素的时候啊.

那么下面的集合 A, B, C 等不等?

$$A = \{B\}, B = \{A\}, C = \{C, A\}$$

经典的集合论不容许出现这种集合 (不容许如下的无穷链):

$$\cdots \in X_3 \in X_2 \in X_1 \in X_0$$

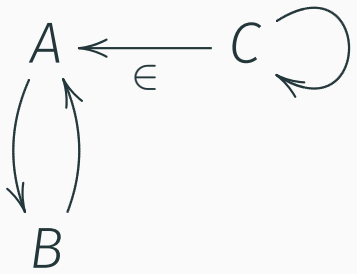
但是我们还是可以定义它们间的等价关系 (看成是相等的):

直观: 集合 x 和 y 被认为是等价的则至少如下的应该成立:

- 对每个 x 的元素 x' 都有 y 的元素 y' 使得 x' 和 y' 等价,
- 对每个 y 的元素 y' 都有 x 的元素 x' 使得 x' 和 y' 等价.

非良基集合论与互模拟 A={B}, B={A}, C={C,A}

我们可以用图的方式看的更清楚 (用箭头表示集合和其元素的关系):



注意, 在看 A 和 B 是否等价的时候还会用到 A 和 B 是否等价... 是不是还是循环了? 既可以等价又可以不等价...

一个图里面点之间的非空二元关系 Z 被称为是一个互模拟关系 (bisimulation) iff 它满足对**所有** xZy 的 x, y 有:

- 对所有 x', 如果 xRx' 则有 y' 使得 yRy' 且 $x'Zy'$,
- 对所有 y', 如果 yRy' 则有 x' 使得 xRx' 且 $x'Zy'$,

一个图里两个点 x, y 是互模拟的 (bisimilar) iff **存在**一个互模拟关系 Z 使得 xZy. 上面的 A, B, C 都是互模拟的, 都可以看成是等价的.

克里普克模型上的互模拟

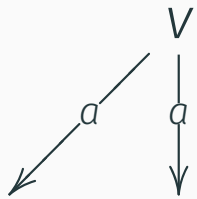
一个非空的在两个克里普克模型 $\mathcal{M}, \mathcal{N}$ 的可能世界之间的二元关系 Z 是一个互模拟关系 iff 它满足对**所有** xZy 的 x, y 有:

- x 和 y 上满足的基本命题一样.
- 对所有 x', 如果 xRx' 则有 y' 使得 yRy' 且 $x'Zy'$,
- 对所有 y', 如果 yRy' 则有 x' 使得 xRx' 且 $x'Zy'$.

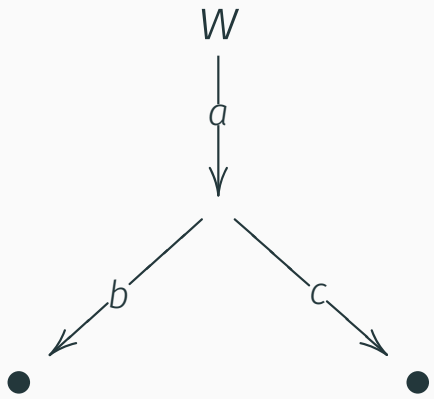
两个点模型 $(\mathcal{M}, w)$ 和 $(\mathcal{N}, v)$ 是互模拟的 $(\mathcal{M}, w \leftrightarrows \mathcal{N}, v)$ iff 存在 $\mathcal{M}$ 及 $\mathcal{N}$ 上的一个互模拟关系 Z 使得 $(w, v) \in Z$.

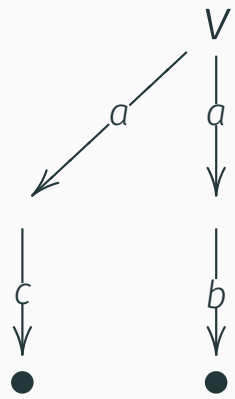
$Z = \{(w, v), (2, v), (1, 3)\}$ 是一个互模拟关系, 所以 $\mathcal{M}, w \leftrightarrows \mathcal{N}, v$.

互模拟与模态逻辑等价关系的对应



模态不可区分的, 确实是互模拟的.





模态可区分的, 确实不互模拟.

在有穷点模型上 (可推广到几乎所有模型), 模态等价关系和互模拟是一样的! 其中一个方向不需要汾河限制: 只要互模拟, 就一定模态不可区分.

游戏的角度看互模拟:

给定点模型 M, w 和 N, v: 定义游戏如下:

- 游戏者: 支持者 (Proponent) 和反对者 (Opponent) P 认为俩模型 “一样”, O 认为不一样.
- 游戏的状态 $\langle w', v' \rangle$ 两点分别在 $\mathcal{M}, \mathcal{N}$ 中, 初始状态是 $\langle w, v \rangle$.
- 游戏的规则:
  - O 选 w' 或者 v' 的某个 a-后继 x,
  - P 需要对上另一点的 a-后继 y, 并且相应改变游戏状态并继续.
- 游戏的输赢条件: O 赢 iff
  - 当前状态两个点的基本命题赋值不同, 或者:
  - P 没得走了.

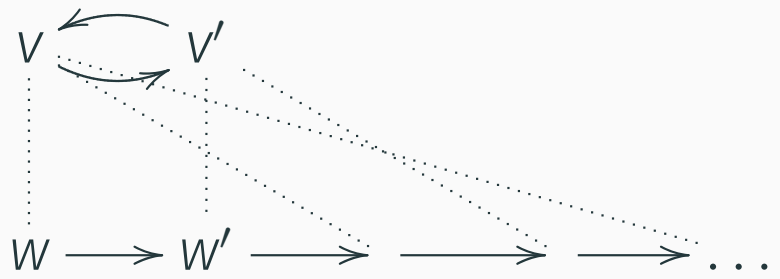
可以证明: P 在 $\mathcal{M}, w$ 和 $\mathcal{N}, v$ 上的互模拟游戏里有必胜策略 iff $\mathcal{M}, w \leftrightarroweq \mathcal{N}, v$. 这个游戏还可以限制在有穷多轮上.

互模拟在逻辑里的应用

压缩模型, 证明 (不) 可定义的性质, 刻画表达能力.

回到之前的问题, 如何证明反对称性 (两点互相通达则是一个点) 是不可模态定义的?

要证明: 任给 模态公式 φ 并非对任意框架 $\mathcal{F}$ 都有 $\mathcal{F} \vDash \varphi \iff \mathcal{F}$ 是反对称的. 证明: 假设有这么个公式 φ 可以定义反对称性, 考虑下面两个框架 (下 $\mathcal{F}$, 上 $\mathcal{F}'$), 根据假设 $\mathcal{F} \vDash \varphi$, 但 $\mathcal{F}' \nvDash \varphi$:



这意味着 $\mathcal{F}'$ 有个赋值 V' 还有个点 x, 使得 $\mathcal{F}', V', x \nvDash \varphi$, 但是按照虚线我们可以把 V 这个赋值迁移到 $\mathcal{F}$ 上 (记为 V),还使虚线是一个互模拟关系, 所以 $\mathcal{F}', V', x \leftrightarroweq \mathcal{F}, V, y$,因而 $\mathcal{F}, V, y \nvDash \varphi$, 与 $\mathcal{F} \vDash \varphi$ 矛盾. 所以反对称性不可定义.

循环定义很普遍, 计算机定义无穷串 $w = aw$, 自然语言的例子.

Merriam-Webster 词典 (2013):

- Hill (1): “a usually rounded natural elevation of land **lower than a mountain**”
- Mountain (1a): “a landmass that projects conspicuously above its surroundings and is **higher than a hill**”

- Oak (1a): “any of a genus (Quercus) of trees or shrubs of the beech family that produce **acorns**”
- Acorn: “the nut of the **oak tree**”

我们很多时候其实还蛮能理解这些循环的...

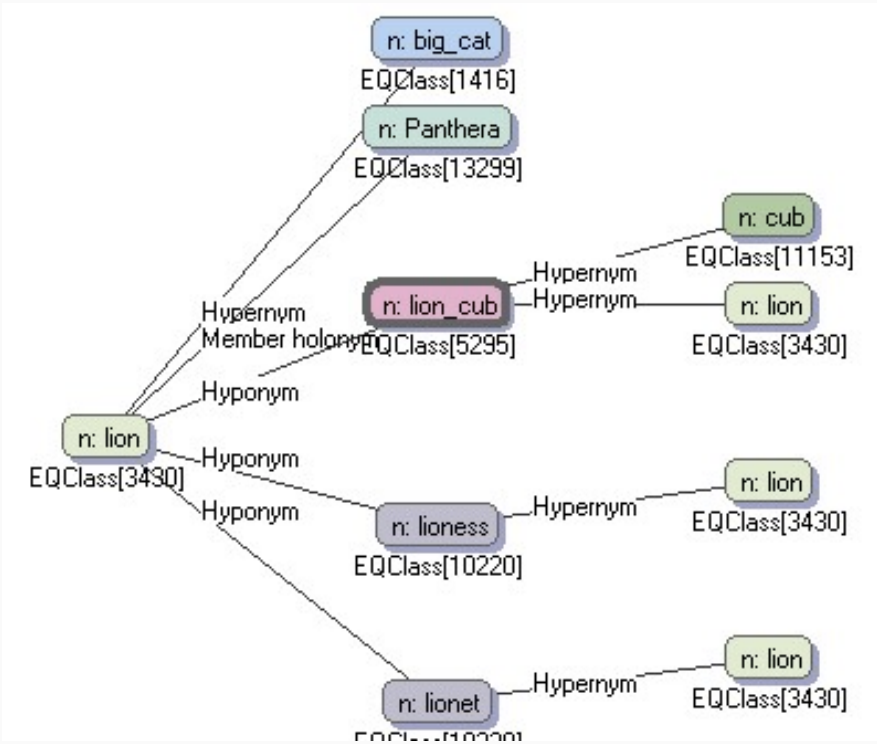
本质上你与别人的关系也很大程度上决定你的性质. 在中国, 你是谁没关系, 你爸是谁或者你老板是谁似乎更重要....

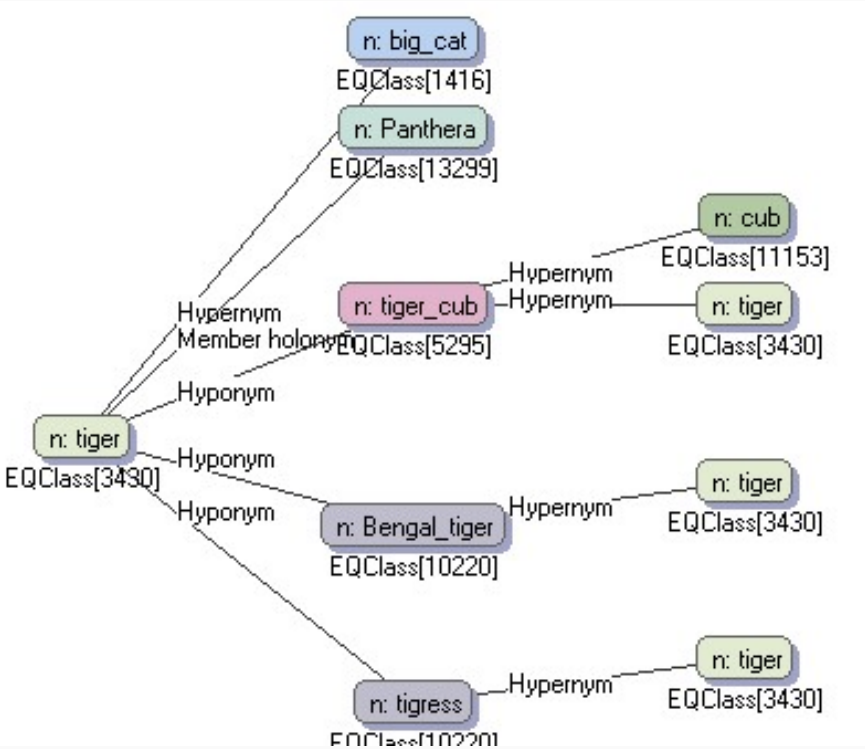
WordNet as a (big) Kripke model

WordNet (Miller, Fellbaum) 3.0 database contains 155,287 words organized in 117,659 synsets (sets of synonyms) for a total of 206,941 word-sense pairs.

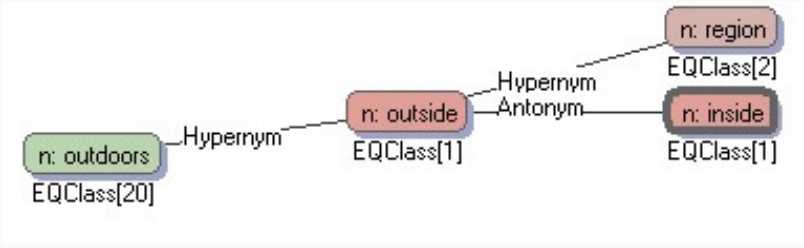
- Synsets as nodes
- Semantic and lexical relations as labelled relations between nodes: hypernyms, meronym, antonyms, etc
- Categories and types of words as the properties on the nodes: nouns, verbs, adjectives, adverbs and so on.

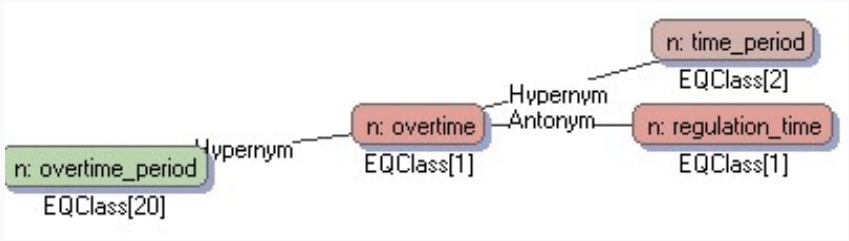
计算 WORDNET 上互模拟的点 (与熊伟合作)

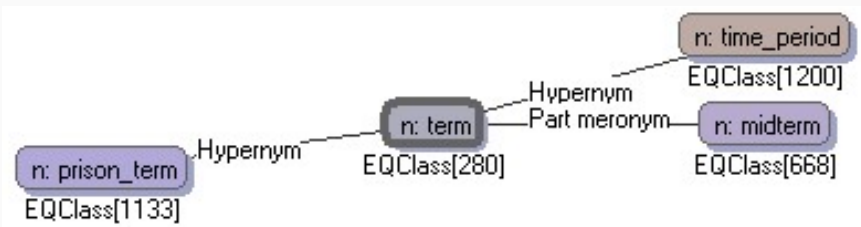


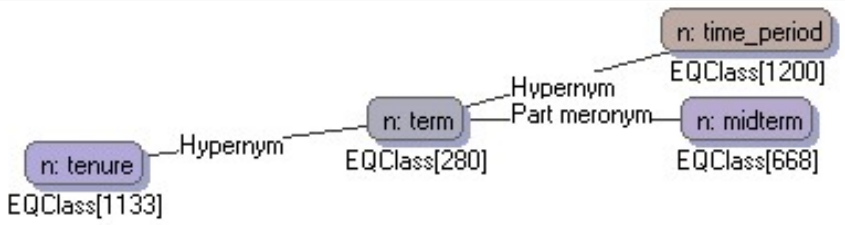


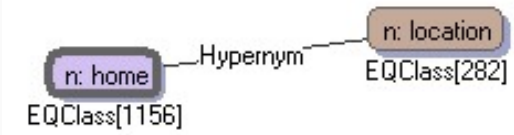
计算 WORDNET 上互模拟的点

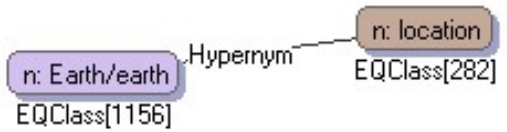




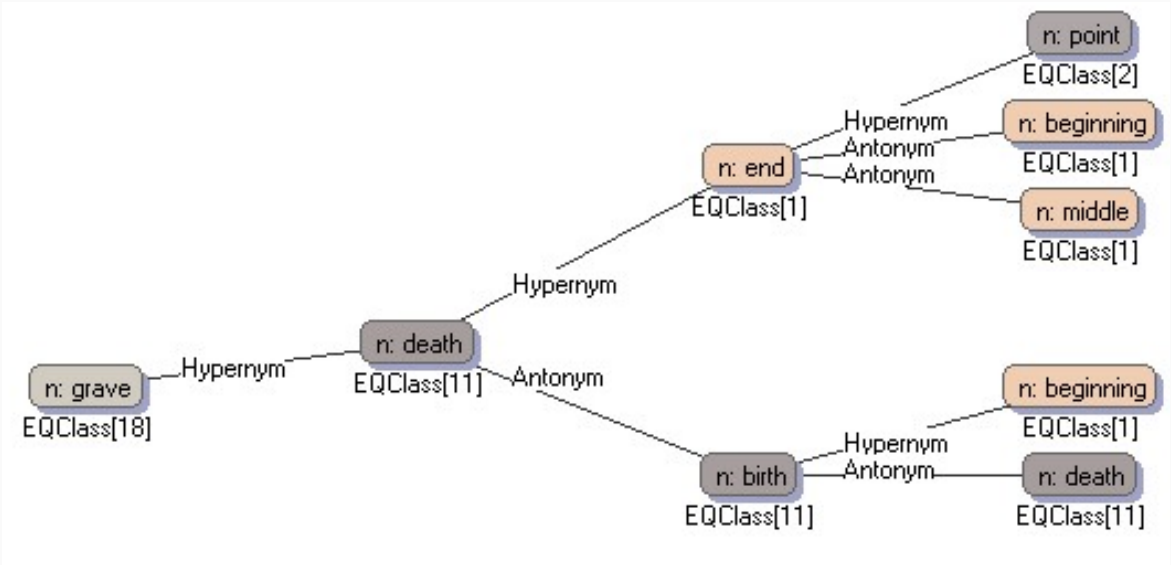


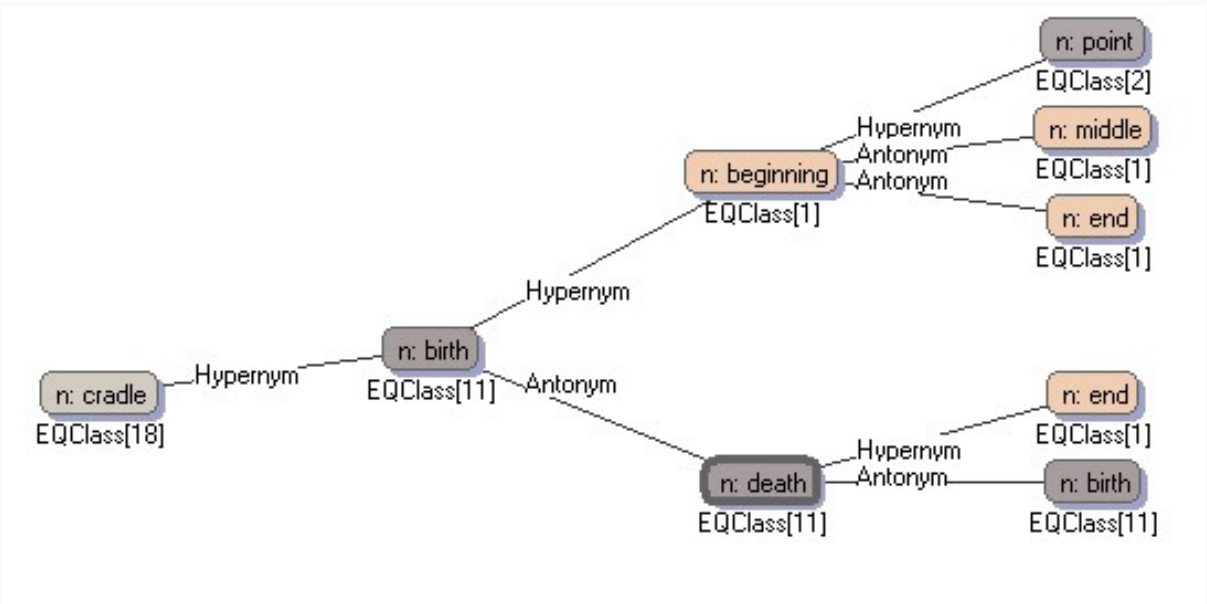






计算 WORDNET 上互模拟的点





也许可以帮助我们处理类比和隐喻. 相似性不等于文献的共现性.

(余) 归纳及其他

互模拟的定义的特点

- 先给个关系再局部性的检查
- 不是一个归纳 (inductive) 的定义而是一个余归纳 (coinductive) 定义
- 特殊的余归纳证明方法

归纳定义: 比如我们的逻辑语言, 什么是公式, 如果什么是公式, 那么什么也是公式, 而且也就是这么生成的是公式, 其他不是.

那余归纳是啥? 为什么可以这么定义?

归纳和余归纳背后的技术

归纳:

- 从最小一点点造上去
- 定义出来的集合 (记为 $X^\star$) 其实是不等式 $f(X) \subseteq X$ 的最小解
- 我是则什么也是 (封闭性)
- 证明 X^* 里的都有性质 Y 的办法: $f(Y) \subseteq Y \implies X^\star \subseteq Y$.

余归纳:

- 从全集一点点排除下来
- 定义出来的集合 (记为 $X^\star$) 是不等式 $X \subseteq f(X)$ 的最大解
- 有什么和自己有关的性质
- 证明 Y 里的也是 X^* 的办法: $Y \subseteq f(Y) \implies Y \subseteq X^\star$.

事实上余归纳定义 X 可以看作归纳定义 X 的补.

例子

定义模型中从 w_0 可通达到的点的集合,其实就是 $f(X) \subseteq X$ 的最小解:

$$f(X) = \{w \mid w_0 R w\} \cup \{w \mid \exists v \in X : vRw\}$$

定义模型中有无穷路径的点的集合, 其实就是 $X \subseteq f(X)$ 的最大解:

$$f(X) = \{w \mid \exists v \in X : wRv\}$$

也可以归纳定义其补集: 死点不是, 只能通达到不是的也不是...

对互模拟来说, $f_{\leftrightarrows}(Z) = \{(x, y) \mid x, y$满足那三条关系 (基于 $Z)\}$, Z 是一个互模拟关系 iff $Z \subseteq f_{\leftrightarrows}(Z)$. 证明 (x, y) 是互模拟的 (其实是所有互模拟关系的并), 就是证明他们在某个互模拟关系里.

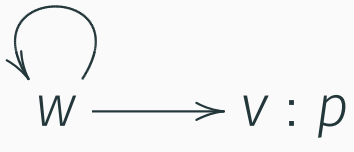
最小最大解的存在性通常需要函数是单调的, 由 Knaster-Tarski 不动点定理保证 (幂集上的单调函数有最小最大不动点就是相应不等式的最小最大解). Kleene 不动点定理告诉你应该怎么算.

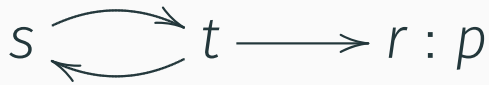
有穷模型里找互模拟关系

可以从最大的全关系不断的迭代缩小直到停止:

$$Z_0 = W \times W, Z_1 = f_{\leftrightarrows}(Z_0), Z_2 = f_{\leftrightarrows}(Z_1), \ldots$$

Example (让我们只看模型间的点对儿)





$\{(w,s),(w,t),(w,r),(v,s),(v,t),(v,r)\}$
$\{(w,s),(w,t),(v,r)\}$
$\{(w,t),(v,r)\}$
$\{(v,r)\}$

归纳定义的也有有趣的东西: 再来看我们的形式语言

比如命题逻辑语言 (归纳定义的):

$$\varphi \quad ::= \quad p \mid \neg\varphi \mid (\varphi \wedge \varphi)$$

从语言学的角度, 生成这个语言的叫上下文无关语法 (context-free grammar CFG), BNF 是一种在计算机里写这样语法的方式.

语法规则 (production rules):

- $S \implies p \ (p \in \mathbf{PV})$
- $S \implies \neg S$
- $S \implies (S \wedge S)$

这些语法规则生成的语言 (不带 S 的符号串集) 就是能从 S 使用语法规则推出来的不带 S 的那些表达式. 更一般的, 给定非终结符 (non-terminal symbols) 的有穷集合 N, 终结符 (terminal symbols 实际表达式里的符号) 的有穷集合 Σ, 一个 CFG 规则的形式是 N 里某非终结符 $A \implies$ 一个由 N 和 Σ 里符号组成的有穷的符号串

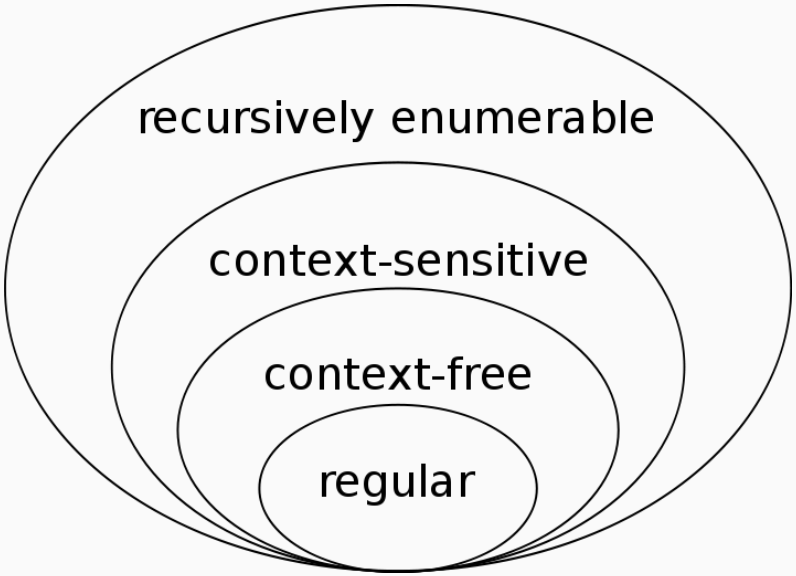
更复杂的语法

上下文相关语法 (Context-sensitive grammar), 比如:

- $S \Longrightarrow aBC$
- $S \Longrightarrow aSBC$
- $CB \Longrightarrow CZ$
- $CZ \Longrightarrow WZ$
- $WZ \Longrightarrow WC$
- $WC \Longrightarrow BC$
- $aB \Longrightarrow ab$
- $bB \Longrightarrow bb$
- $bC \Longrightarrow bc$
- $cC \Longrightarrow cc$

这个语法生成语言 $\{a^n b^n c^n \mid n \geq 1\}$.

乔姆斯基 (CHOMSKY) 的形式语言分层



文法	形式语言	计算模型	产生式规则
0-型	递归可枚举语言	图灵机	$\alpha \Longrightarrow \beta$（无限制）
1-型	上下文相关语言	线性有界非确定图灵机	$\alpha A \beta \Longrightarrow \alpha\gamma\beta$
2-型	上下文无关语言	非确定下推自动机	$A \Longrightarrow \gamma$
3-型	正则语言	有限状态自动机	$A \Longrightarrow aB$
			$A \Longrightarrow a$

A 是非终结符, α, β, γ 是可以有终结符和非终结符的有穷串.

正则表达式和有限状态自动机

正则表达式 (regular expressions) ($a \in \Sigma$)

$$\pi ::= a \mid \pi \cdot \pi \mid \pi + \pi \mid \pi^*$$

定义语言每个正则表达式对应的形式语言:

- $L(a) = \{a\}$
- $L(\pi \cdot \pi') = \{wv \mid w \in L(\pi), v \in L(\pi')\}$
- $L(\pi + \pi') = L(\pi) \cup L(\pi')$
- $L(\pi^*) = \bigcup_{n \geq 1} L(\pi^n) \cup \{\epsilon\}$

ϵ 是空串.

正则表达式和有限状态自动机

一个有限状态自动机 $\mathcal{A}$ 是 $(Q, \Sigma, q_0, \delta, F)$

- Q 有穷的非空状态集, $q_0 \in Q$ 是初始状态;
- Σ 是一个有穷的符号集;
- $\delta \subseteq Q \times \Sigma \times Q$ (表示状态转换关系);
- $F \subseteq Q$ 是接受状态集.

我们说 $\mathcal{A}$ 接受 Σ 中符号的有穷串 σ iff 从 q_0 开始可经过 σ 里符号所代表的状态转换后停在某个接收的状态上. 一个自动机所代表的语言 $L(\mathcal{A})$ 是所有这个自动机接受的 Σ 里字符的有穷串的集合.

可证下面三个等价 (给定符号表 Σ):

- 正则语法能定义的语言们
- 正则表达式所能定义的语言们
- 有穷自动机能接受的语言们