形式语义学读书汇报手稿第一版

Comma

2025.10.26

目录

1 前情回顾

我们在上一节课里学习了句法学（syntax）、形态学（morphology）以及形态句法（morphosyntax）的相关问题。我们知道我们语言学的研究目标要达到三个充分性（Adequacy），即观察充分性、描写充分性以及解释充分性，尤其是解释充分性，这是理论构建的最高要求。如果我们说生成语法（Generative Grammar）要达到解释充分性，或者说正在达到解释充分性，那么我们的一个核心问题是，解释的是什么，也就是什么需要被解释？那么根据上一节课的内容，我们知道乔姆斯基真正想要解释的并不是语言的 performance（表现）层面，而是 competence（能力）层面，即内嵌于我们大脑中的 knowledge，也就是说我们为什么会有某种 performance，以及儿童何以在刺激贫乏（Poverty of the Stimulus）的条件下快速习得（Acquisition）语言，究其本质是因为我们先验性地具有一种需要被理论反推出来的 knowledge，而这种 knowledge 则通过人类特有的语言直觉（Intuition）表征

出来。而这种直觉是什么呢，回顾上一节课所学，在句法（syntax）的层面，主要有三种直觉，第一是我们能够判断句子的合语法度（Grammaticality），即句子合不合语法；第二是我们能够判断句子的可接受性（Acceptability），即我们能够对那些不合语法的句子，进行可接受性的排列；第三是我们能够判断句子的结构歧义（Ambiguity），比如“咬死了猎人的狗”，我们知道这一句子的歧义主要是由于句法结构导致的，这种对于结构歧义的敏感也属于我们语言直觉的范畴。

具体细节，大家可参考:

乔姆斯基 1975 年 the logical structure of Linguistic theory 与 1965 年 Aspects of the Theory of syntax。

既然在句法的层面，我们表现出了一种能够反映语言 knowledge 的直觉，那么，在语义（semantics）层面上呢，理论上讲，如果我们承认我们的语言能力也包括语义解读的话，是不是说明语义也应该具有某种和句法类似的语言直觉呢？那么这种直觉是什么？正是我们能够理解句子的意义，比如“雍正在月球上听张程鑫讲语言学如何救清朝”，这句话毫无疑问是不现实的，我们为什么觉得此句荒谬，因为此句的语义确实被我们理解了，我们根据各自的经验能够想象这样一个荒谬的场景，所以我们说我们也具有语义层面的语言直觉，因为我们可以根据某种方式理解语义。既然我们从本质上无法排除这种语义直觉，那么我们就必须把它纳入到我们宏伟的解释蓝图之中，即我们现有的理论必须解释这种理解语义的能力，才能向解释充分性的高级目标更进一步。

我们再回到上节课的内容当中，来看一看坚持句法核心主义的以乔姆斯基本人为代表的句法学家是如何理解语义部分的，他们在 80 年代管约论（GB）中指出了这样的倒 Y 模型：

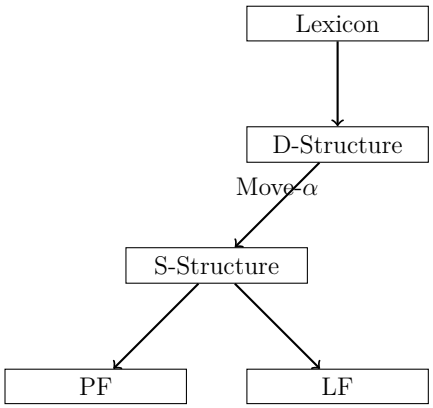


图 1: GB 倒 Y 字模型

这一模型为我们展示了语言生成的完整程序，即先生成深层（deep）结构，再通过移位（Move）的操作，生成表层（surface）结构，最后将表层结构输送到音系和语义的界面，进行拼读与解读，而语义的解读正是在 syntax-semantics 的界面上进行操作的，这就说明在生成语法中，语义的解读与句

法的推导密不可分，我们甚至可以回到生成语法的标准理论时期，那时乔姆斯基认为语义解读是由深层结构决定的，总而言之我们是无法抛开句法结构谈语义解读的。那么我们不由想问，语义的解读到底是如何操作的，其与句法的关系到底是怎么样的？

因此我们在这里提出了两个问题：

第一，我们的语义直觉或我们为什么能够理解语义这一问题该如何在理论中进行解释？

第二，语义解读是如何进行的，以及其与句法有什么关系？

当然，我们伟大的先辈已经为我们指明了方向，也就是我们今天要重点介绍的内容，形式语义学！

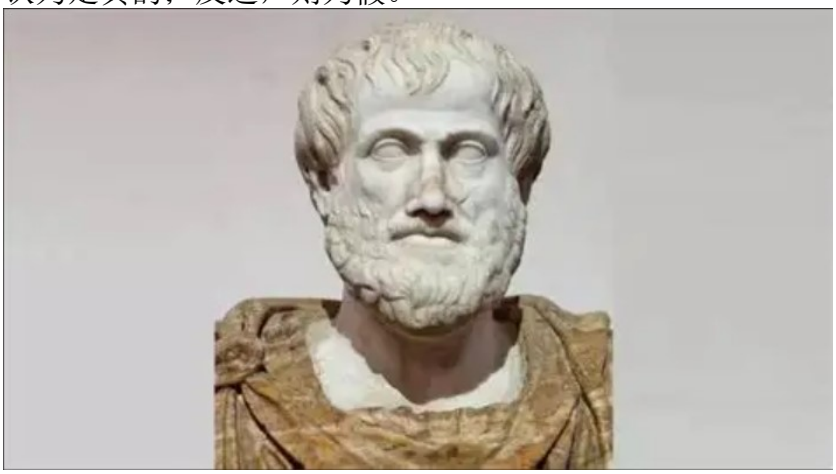
参考教材：

Semantics in Generative Grammar (Irene Heim, Angelika Kratzer)

形式语义学引论 (蒋严 (Yan JIANG), 潘海华 (Haihua PAN）)

2 基于真值条件（Truth-conditional）的语义学

首先，什么是真值？真值是值，所以它绝对不等同于真。真值这一概念最早可以追溯到亚里士多德，他在《形而上学》中这样写道：“说是者为是，非者为非，即为真；说是者为非，非者为是，即为假”这意味着，真理的本质在于陈述与现实之间的一致性，只有当一个陈述与事实相符时，它才被认为是真的；反之，则为假。



真值的概念后来经由弗雷格的发展，到塔斯基才被严格定义, 塔斯基严格区分了对象语言和元语言，前者是我们正在谈论其语句真值的语言，例如我们这里主要谈论英语和汉语，后者是用来谈论对象语言的语言，比如我们如果用英语谈论英语的语句真值，那么英语就是英语的元语言。进而塔斯基通过递归与满足定义了“为真”这个谓词，他通过以下步骤为一个特定的形式化语言（如类演算）定义真值：(注意：此处无需掌握，只是留给感兴趣的同学自行了解该内容，详细内容可参考塔斯基选集：Alfred Tarski: Collected Papers. 以及 Alex Malpass, Marianna Antonutti Marfori：The History of Philosophical and Formal Logic: From Aristotle to Tarski)

列出初始项：明确对象语言的所有基本符号（如变量、常量、谓词、逻

辑连接词 $\wedge, \vee, \neg, \rightarrow$ 和量词 $\forall, \exists$）。

递归定义“满足”：

一个开公式（包含自由变量的公式，如“x 是白的”）本身没有真值。

但一个对象序列（例如，给变量 x 赋值“雪”）可以满足这个公式。

他从最简单的原子公式开始，递归地定义了所有复杂公式的“满足”条件：

原子公式：序列满足 F(x)，当且仅当，赋给 x 的对象具有性质 F。

合取：序列满足 $(A \wedge B)$，当且仅当，它满足 A 并且满足 B。

否定：序列满足 $\neg A$，当且仅当，它不满足 A。

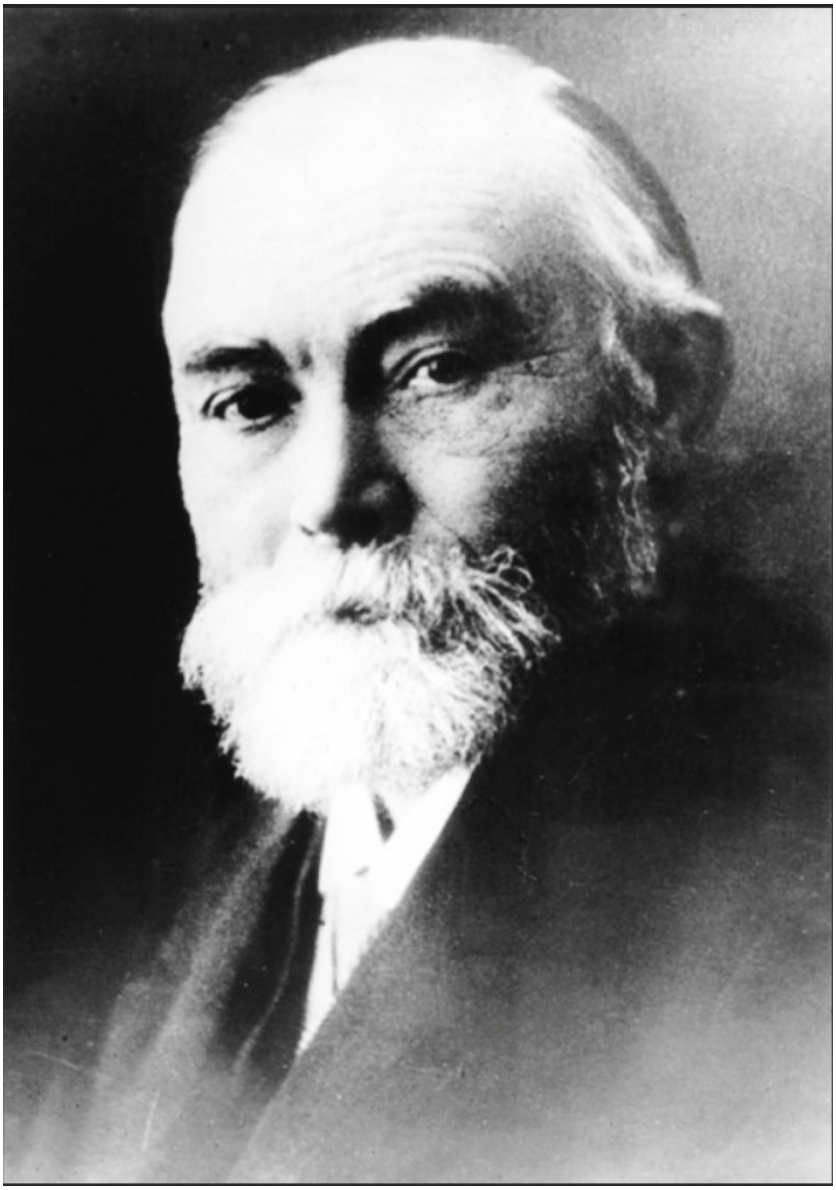
全称量词：序列满足 $\forall x\ A(x)$，当且仅当，所有将 x 赋值为某个对象的序列都满足 A(x)。

由“满足”定义“真”：

一个闭语句（没有自由变量的公式）要么被所有对象序列满足，要么被没有对象序列满足。

定义：一个闭语句是真的，当且仅当，它被所有对象序列满足。

定义：一个闭语句是假的，当且仅当，它不被任何对象序列满足。



但是这些严格的数学定义距离我们太过遥远，理解起来较为费力，我们干脆采用《形式语义学引论》中的简单“定义”，即“一个陈述性的句子或者真实地反映了外部世界的某个现象，或者对某现象做出了不正确的、虚假的描述，用逻辑的术语说，前者为真（true），后者为假（false）。真和假统称为真值（truth value）。”因此我们需要搞懂的第一个重要的概念即为真值，也就是真和假的统称，即真值有两个取值，要不是真，要不是假。英国数学家乔治·布尔（诺奖获得者辛顿的曾祖父）用代数的形式对真值进行表征，即 1 代表真，0 代表假，所以我们可以把真值看成一个集合，即只有 0 和 1 或假和真两个元素的集合。

理解了真值的概念，我们可以进一步研究句子的真值条件（truth condition），即研究一个句子在什么条件下为真，什么条件下为假，或者我们可以说，我们是通过解读一个句子的真值条件来解读句子的语义的，当我们知道一个句子在什么条件下为真，在什么条件下为假时，我们就说我们从真值条件这一角度解读了句子的语义，注意我们并不是说句子的语义等于其真值条件义，只是说解读真值条件是解读语义的一个重要方面。因此形式语义学主要是基于真值条件的语义学，它承认我们应该将真值条件与语义解读勾连起来，那么我们如何获得句子的真值条件呢？换句话说，在基于真值条件的形式语义学的框架下，我们如何对语义进行解读呢？试看下面的句子：

书包里装着一盒美味的炸药。

在弗雷格的框架中，这一句子的真值条件是：

句子“书包里装着一盒美味的炸药”为真，当且仅当书包里装着一盒美味的炸药。

我们可以将此例进行推广，得到 Schema for truth-conditions：

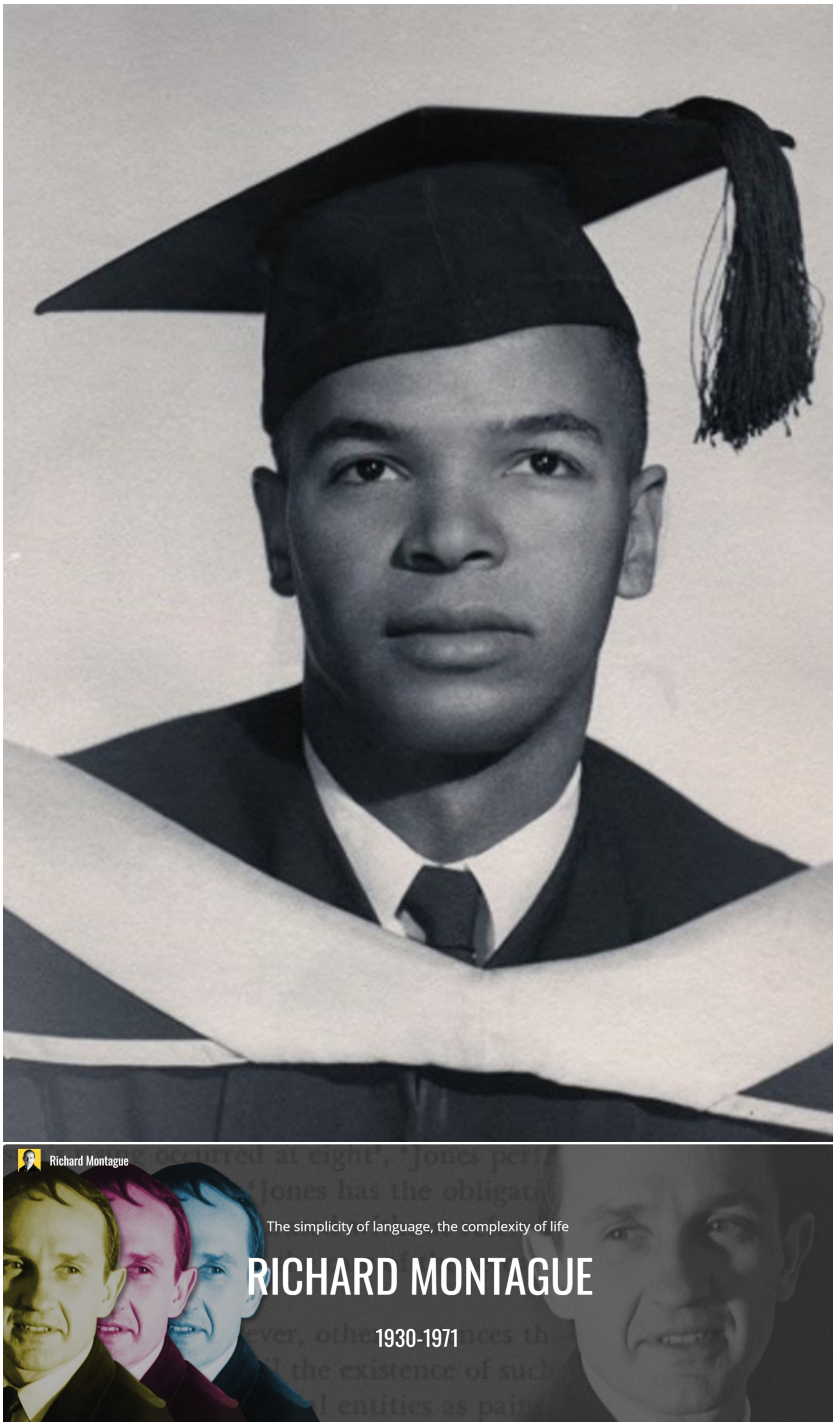
The sentence “X” is true if and only if X。（这里请大家注意对象语言和元语言的区分）

显然，即使我们假设这一推广是正确的，它对于我们解读语义没有显著的帮助，更不能解释我们的语义直觉，也无法解释自然语言的创造性，注意，我们是语言学专业，所以需要在语言学的语境之下，探讨这一问题。

因此，我们需要更换一种方案，这一方案是：整个句子的意义是由部分（parts）的意义通过某种方式计算出来的，即我们需要组合性（composition）原则:

Every meaningful part of a sentence contributes to its truth-conditions in a systematic way.

即每一个具有意义的部分都须以一种系统的方式作用于一个句子的真值条件，也就是说，整体的意义等于部分的意义通过严格的系统组合之后得到的意义。而这种严格的系统组合某种程度上正是我们的句法结构，因此按照著名数理逻辑学家、形式语义学祖师爷理查德·蒙太古（Richard Montague，塔斯基的学生，也是塔斯基最大的反对者，塔斯基认为自然语言和人工语言在理论上有本质的区别，但是蒙太古则认为自然语言与人工语言间并没有任何重要的理论性区别，他认为可以通过一个自然而精确的数学理论来理解这两种语言的语义与句法 (Montague 1970c, 222)，即在乔姆斯基把广义的语法还原到心理学之后，蒙太古试图将其还原至数学上）的观点，句法的完备性（well-formedness）和语义的可解释性（interpretability）在某种程度上是相吻合的，即句法的推导和语义的解读属于一种同态运算，具体体现为所有句法成分均需获得语义解释。总之，我们的形式语义学不仅是基于真值条件的，同时也是基于组合性原则的。


Richard Montague
The simplicity of language, the complexity of life
RICHARD MONTAGUE
1930-1971

这里，我们可以用弗雷格的组合性原则来理解，弗雷格进行了一个 conjecture（推测），即他推测语义的组合总是在于对一个不饱和的意义成分的饱和化（the saturation of an unsaturated meaning component），即可以理解为语义的组合是函数（function）的应用（apply）。例如下面的句子：

Caesar conquered Gaul.

在这个句子中，Caesar 是一个论元（argument），（注意，在形式语义学中，可以把这个论元理解为高中数学中的自变量 x）而动词短语 conquered Gaul 则是一个函数，它是不饱和的，因为它还需要一个论元成分才能表达完整的意义（饱和化），所以，这个句子完整意义的表达，是将后半部分的函数应用于前半部分的论元，然后输出或者返回一个真值。（注意，在形式语义学中，我们一般说把函数 apply 在一个论元上，也就是说不能理解为把一个论元放在函数中进行运算，大家需要慢慢地适应这种思维。）

通过弗雷格的这个例子，我们可以初步地理解什么叫组合性原则。具体细节我们将在后面逐渐展开。

这里请同学们注意，我们这里探讨的形式语义学是较为狭义的概念，即生成语法中的形式语义学，有时候会称其为“generative semantics”，但是我们不能将其翻译为生成语义学，因为是存在一个专门的语义学分支就叫生成语义学，此生成语义学与我们所探讨的相去甚远，有本质的区别，所以请大家不要混淆，谢谢！

3 逻辑演算基础（了解性内容）

在正式开始介绍形式语义学的基本理论和技术细节之前，我们需要回顾我们曾经接触过的一些简单的数学基础，其中大家一定要理解的是两个基础概念，一是集合（set），二是函数（function）。

注意，由于这些纯数学部分对于我们来说并没有本质性的帮助，所以我在这一部分中会尽量避免大量冗余的数学符号与相关证明，只是将简单的定义与定理展示给大家，请那些对数学情有独钟的同学宽恕我的这一简化做法，如果想要深入地研究这一部分内容，有能力的同学请自学高等代数，这里推荐大家去 b 站学习北京大学丘维声老师的完整课程。

集合（set）：集合是指具有某种特定性质的具体的或抽象的对象汇总而成的集体。

元素（elements）：构成集合的这些对象叫做元素。

元素与集合的关系是属于 $\in$ 或不属于 $\notin$。

x∈A, 当且仅当 x 是 A 中的一个元素。

一个集合可以是有限的、无限的，也可以是空的，即没有元素，即记作 $\emptyset$。

两个集合相等，当且仅当它们有完全相等的元素。

子集（subset）：一个集合中的所有元素也是另一个集合中的元素。记作 $\subseteq$，若集合 A 是集合 B 的子集，同时集合 B 不是集合 A 的子集，那么我们说集合 A 是集合 B 的真子集，记作 $\subset$。

交集（intersection）：A∩B

并集（union）：A∪B

补集（complement）：$\complement$A

幂集（power set）:$\mathcal{P}(A)$, 即集合的子集组成的集合，包括空集，因为空集是任意集合的真子集。

关系（relation）：

两个集合 A 和 B 之间可以建立某种关系（relation），再设存在 x∈A，y∈B，可以将其构建为有序对（ordered pair）$\langle x, y\rangle$，注意它是有序的，即 $\langle x, y\rangle$ 不等于 $\langle y, x\rangle$，我们把这些有序对组成的集合称为一个二元关系（2-

place relation），并把这种情况称为笛卡尔积，记作 A×B。

而函数（function）就是一种特殊类型的关系，如果定义在集合 A 和 B 上的关系 R 满足以下两个条件，我们称之为从 A 到 B（大家一定要注意顺序）的（全）函数，记作 F：A →B：

a.A 中的每一个元素都只能对应于 B 中的一个元素；

b.A 中的所有的元素都必须对应于 B 中的一个元素。

按照高中数学的说法，我们也可以用 f(x) 来表示一个函数，x 的值取自集合 A，f(x) 的值取自集合 B，我们称 x 为函数的论元（argument），A 为函数的论域或定义域（domain），B 为函数的值域（range），如果不是论域中的所有元素都有一个对应的值在值域中，我们把这种函数叫部分函数（partial function），两个条件都满足的叫全函数（complete function）。

有了集合和函数的概念，我们再回到弗雷格的框架，弗雷格认为语义的组合是一种函数的应用，那么是否每一个语义成分，都有一种函数的表达呢？是！因此就有了解释函数（interpretation function）的概念：

For α, $[\![\alpha]\!]$ is the denotation of α。

其中 denotation 为 α 的指谓，可以理解为它指称的是什么，而这个双中括号则是一个函数，这个函数的功能是，输入一个语义成分 α，输出一个 α 的指谓，所以这个函数又叫解释函数。

因为指谓是和外延（extension）相联系的，所以把基于指谓进行运算的语义学称为外延语义学（extensional semantics），反之，则是内涵语义学（intensional semantics），我们这里暂且不介绍内涵语义学。

解释函数的应用（引入 λ 演算之前）：

$[\![\text{Ann}]\!]$=Ann

$[\![\text{works}]\!]$=f：D→T, 对于所有 x∈D，f(x)=1 当且仅当 x works.

所以这里我们可以粗浅的理解，对于这些专有名词，它们指称的是一

个具体的个体，而对于这些谓词，它们指称的是一个函数。

4 类型论与 λ 演算

4.1 类型论（type theory）

我们再回到上一节课的内容中，我们知道句法在运算过程中，并不是去纯粹运算那些光杆的词，而是需要先定义出句法范畴，也可以理解为给这些词贴上的标签（label）, 在本质上这是一个基于分类的程序，关于句法成分的范畴化与子范畴化可以参考乔姆斯基 1965：Aspects of the theory of syntax。那么我们这里的问题是，既然句法中有范畴，那么语义中是不是也应该对其进行标签式的分类呢？我想这是一个较为合理的类推，确实，在我们的语义学当中，也存在着许多语义类型，它们往往与句法范畴相对应。

首先，我们会想到一些专有的名词，比如张三、李四、Ann 和 John 等等，这些名词都指称一个特定的个体，那么我们可以把此类名词的语义类型定义为 e，即英文 entity（实体），同时我们还会想到一些完整的句子，比如“我热爱形式语义学。”，相比于“爱形式语义学”这类不饱和结构，这一句子具有真值，所以我们把这些表征真值的成分的语义类型定义为 t，即英文 truth（真值）。

所以，聪明的我们已经定义出了两种语义类型，即 e 和 t，因此我们就把 e 和 t 规定为原子类型（primitive types）或者基本类型（basic types），它们是语言单位的逻辑语义范畴，往往与我们学过的句法范畴相对应。由这些基本类型组合起来的逻辑式是复杂类型，相关的理论就是类型论。

我们刚才说一个专指名词的语义类型是 e，而一个完整句子的语义类型是 t，那么我们可以用一个更简洁的方式表达这种说法，即 TP 函数，也叫

做赋类函数（type assignment function）, 其定义域是所有句法范畴组成的集合，其值域是所有语义类型组成的集合，所以我们说：

TP(N 专)=e，以及 TP（S）=t，所以提醒大家在表达语义类型时，尽量不要写成 N 专 =e 或 S=e，可以替换成赋类函数的形式，也可以采用 S：t 这样的形式，同时请大家区分赋类函数与解释函数的区别，即：

TP(S) 不等于 ⟦S⟧, 前者是输入一个语言单位，返回一个语义类型，而后者是输入一个语言单位，返回这个语言单位的指谓。

下面我们来探讨一下其他句法范畴的语义类型，首先对于那些非专指性的名词以及那些充当谓词成分的名词（例如“我是狗”里的狗），例如人、猫、狗、学生、老师、语言学家等，其语义类型并不是 e，因为它们不指称任何一个具体的实体，因此这些语言单位的语义类型是一个 $\langle e,t\rangle$，即一个函数，也就是输入一个实体 e，输出一个真值 t，为什么是函数呢，因为我们前面讲过有序对的概念，$\langle e,t\rangle$ 就是一个有序对，它表征的是一个函数关系，比如对于猫这个词来说，它是一个什么样的函数呢，即给这个函数随意输入一个实体，如果这个实体是猫，那么就输出真值真或真值 1，如果这个实体不是猫，那么就输出真值假或真值 0。

其次，我们可以来探讨动词的语义类型，回顾我们之前所学，我们说，根据动词携带论元的数量，可以把动词分为不及物动词、及物动词和双及物动词。那么对于一个不及物动词来说，它的语义类型是什么呢?

例如句子“张程鑫哭了。”中的“哭”是一个不及物动词，因此它的语义类型肯定不能是 e，因为这一动词本身不专指任意一个实体，同时它的语义类型也不是 t，因为单独的一个“哭”一般情况下不能算作一个有真值的完整句子，那么它只能是一个函数，现在我们的问题变成了，既然它是一个函数，那么谁是它的输入，它又输出了什么呢？在这里，我们说“张程鑫”这个实体 e 是它的输入，而“张程鑫哭了”这个有真值的句子 t 是它的输

出（我们这里不探讨时体范畴的语义，因为我们是基础课），因此我们可以说不及物动词哭是一个语义类型为 $\langle e,t\rangle$ 的函数。

那么，及物动词的情况呢？比如“张程鑫爱李芙孜”(注意，这是一个虚构的句子，大家无需探讨其真实性) 这个句子中的“爱”是一个典型的及物动词，那么我们可以把它也看做一个函数，那么这个函数的输入是什么，输出又是什么呢？直接告诉大家，这个函数把宾语位置的“李芙孜”(语义类型为 e) 作为输入，但是，仅仅输入这一个论元，输出的却是一个不饱和的结构，即“爱李芙孜”, 这是一个没有真值的结构，而是另一个等待应用的函数，把这个函数再应用到另一个论元“张程鑫”上，才输出了一个完整的有真值的句子，因此及物动词的语义类型是 $\langle e,\langle e,t\rangle\rangle$，它输入一个实体 e，返回一个函数 $\langle e,t\rangle$，这个函数再输入一个实体 e，才返回一个真值 t，这对应了及物动词带有两个论元的句法特性。

以此类推，我们得到了以下句法范畴的语义类型：

TP(S)=t;

TP(N 专)=e;

TP(N 普)= $\langle e,t\rangle$;

TP(V 不及物)=$\langle e,t\rangle$;

TP(V 及物)=$\langle e,\langle e,t\rangle\rangle$;

TP(V 双)=$\langle e,\langle e,\langle e,t\rangle\rangle\rangle$;

TP(Adj 谓词，例如我帅里的帅)=$\langle e,t\rangle$;

TP(修饰普通名词的 Adj)=$\langle\langle e,t\rangle,\langle e,t\rangle\rangle$;

TP(修饰动词短语的 Adv)=$\langle\langle e,t\rangle,\langle e,t\rangle\rangle$;

练习：

TP(修饰句子的 Adv)=?

注意，语气助词、感叹词等与句子真值无直接联系，所以没有语义类

型。

那么分出语义类型的价值是什么，我们来看一下这个例句：

昨天张三很快地看完了《形式语义学引论》。

截止到现在，以我们的知识，我们可以用类型论分析这一句子的语义，首先，昨天是一个修饰整个句子的副词，其语义类型为 $\langle t,t\rangle$, 张三是一个具体实体，其语义类型是 e，很快地是一个修饰动词短语的副词，其语义类型为 $\langle\langle e,t\rangle,\langle e,t\rangle\rangle$，看是一个及物动词，所以其语义类型是 $\langle e,\langle e,t\rangle\rangle$，《形式语义学引论》是一个具体的实体，所以其语义类型是 e。那么我们就可以像搭积木一样组合这些语义类型，看看这些语义类型如何组合最终才能输出一个语义类型为 t 的句子。

首先，及物动词看（了）和《形式语义学引论》结合，后者相当于前者的论元，所以把函数应用到论元上，返回一个输出值，即另一个函数，所以整个 VP 短语的语义类型就是 $\langle e,t\rangle$，然后，这个 VP 再与修饰性成分很快地结合，因为很快地的语义类型为 $\langle\langle e,t\rangle,\langle e,t\rangle\rangle$，所以可以将这一函数应用到 VP 上，VP 作为输入，返回一个输出值，即又一个函数 $\langle e,t\rangle$，然后，再与论元张三结合，即把这一新的函数应用到论元张三上，张三 e 作为输入，返回一个输出值，即真值 t，这个真值 t 再进一步与修饰句子的副词昨天组合，即把这个副词所指谓的函数应用到论元上，这时真值 t 作为输入，返回一个输出值 t，这就得到了整个句子的语义类型 t。我们刚才的推演，实际就是按照组合性原则来计算的，当然组合性原则的详细内容不是我们这一章节的任务。

完整过程就是：

$\langle e,\langle e,t\rangle\rangle$(e)=$\langle e,t\rangle$

$\langle\langle e,t\rangle,\langle e,t\rangle\rangle(\langle e,t\rangle)$=$\langle e,t\rangle$

$\langle e,t\rangle$(e)=t

$\langle t,t\rangle$(t)=t

大家一定要理解函数的应用是什么意思，我个人认为还是用输入——输出的概念来理解比较好。

4.2 λ 演算

好的同学们，我们刚刚花了些时间深入探讨了形式语义学的类型论。我们了解到，类型论不是一个任意的规则集合，而是一个强大的符号系统。

让我们快速回顾一下类型论为我们解决了哪些核心问题：

a. 防止范畴错误：它确保我们不会写出像“喝水吃了形式语义学”这样无意义的表达式。

b. 明确函数范畴：它告诉我们，“喜欢”这个函数需要什么类型的输入（比如两个实体 $\langle e,\langle e,t\rangle\rangle$），以及会输出什么类型（一个真值 t）。

c. 为组合性提供架构：它保证了当我们把较小的、有类型的表达式组合成更大的表达式时，最终能产生一个有意义的命题（类型 t）。

但是，现在我想请大家思考一个更深层次的问题：

类型论就像一个极其严格的建筑规范——它告诉我们房子需要地基、墙壁和屋顶，并且它们必须按照某种方式连接。它确保了结构的稳固性。

然而，这个规范本身，并没有告诉我们墙壁具体应该用什么材料建造，或者屋顶的确切形状是什么。

换句话说：

我们知道“张三”的类型是 e。

我们知道“爱”的类型是 $\langle e,\langle e,t\rangle\rangle$。

我们也知道“形式语义学”的类型是 e。

我们知道当我们将 $\langle e,\langle e,t\rangle\rangle$ 应用于 e 时，会得到一个 $\langle e,t\rangle$, 再将 $\langle e,t\rangle$

应用于 e 时，得到一个 t。但是，“爱”这个表达式本身，其内在的、精确的语义结构到底是什么？它内部是如何构造的，以至于当它遇到一个像“形式语义学”这样的类型时，就能产生“对所有个体 x，如果 x 是张三，那么 x 爱形式语义学”这样的真值条件？

这就是我们目前理论的“黑箱”！

现在，我们需要一个工具来打开这个黑箱，为每一个有类型的表达式赋予一个精确的、可计算的数学内涵。这个工具就是 -演算（lambda calculus）。

它由数学家阿隆佐 • 邱奇在 20 世纪 30 年代首次发表。lambda 演算作为一种广泛用途的计算模型，可以清晰地定义什么是一个可计算函数，而任何可计算函数都能以这种形式表达和求值，包括我们的逻辑语义。

有关兰姆达的严格数学操作此处省略，因为这对于我们来说并不很关键，但是我们还是要考虑纯粹数学异常执着的同学们，因此我建议此类同学可以去各知识型网站搜索，例如知乎、CSDN 等, 都有较为完整的讲演。我们这里直接来看下面的转换：

⟦smokes⟧=[λx:x∈D.x somkes]

这是一个从解释函数转换为 λ 函数的实例，其中 x 是这一函数的输入，或者说论元，其定义域为 D，D 指的是 De，即真实世界中所以存在的个体组成的集合，还有的定义域是 D$\langle e,t\rangle$, 即表示所有语义类型为 $\langle e,t\rangle$ 的元素组成的集合，以此类推，而这一函数的应用结果，或者说函数体，则是 x smokes。所以这个兰姆达函数将每一个在 D 中的 x 映射（map）到 1（真值为真），当且仅当 x smokes，反之，则将 x 映射到 0（真值为假）。

我们可以再看一个实例：

⟦loves⟧=[λx:x∈D.[λy:y∈D.y loves x]]

这个函数表达了这样的意思，即这个函数把所有在 D 中的 x 映射到了另一个函数中，而此函数又将所有在 D 上的 y 映射到 1，当且仅当 y loves

x，反之，则映射到 0。还记得我们之前说像如 love 这样的及物动词，其语义类型是 $\langle e, \langle e, t\rangle\rangle$ 吗？这里我们用兰姆达函数的形式直观地表达了这一语义类型，即先将函数应用到论元 x（宾语）上，输出了另一个函数，再将这个函数应用到论元 y（主语）上，才输出了真值。

理解了兰姆达函数的意义，我们再来看直接写作过程中，如何对兰姆达函数进行简写。

第一步，省略最外层括号：

当兰姆达表达式不嵌入更大的表达式时，省略其最外层括号，因此我们可以先把上面的表达式简写为：

λx:x∈D.[λy:y∈D.y loves x]

第二步，简化论域条件：

我们可以把论元和其后的论域条件进行合并，形成了：

λx∈D.[λy∈D.y loves x]

第三步，省略默认论域条件：

像如 D 或者 De 是我们默认的论域条件，我们可以省略它：

λx.[λy.y loves x]

我们知道了兰姆达表达式所表达的意思，那么既然这是一种函数形式，我们不由得追问，这一函数如何应用到具体的论元上呢？我们通过下面这个简单的例子来学习一下：

[λx.[λy.y loves x]](Sue)=λy.y loves Sue

[λx.[λy.y loves x](Sue)]=λx.Sue loves x

大家发现规律了吗？

实际上这一规律就是兰姆达表达式的运算法则 β-归约，即假设存在一个兰姆达表达式 f:[λx.M],将 f 应用到论元 N 上,即 f(N)=[λx.M](N)=M[x:=N], 即其应用结果是一个用 N 替换所有 x 的 M。所以大家就记住一句话，消去

λ, 并用新的论元替换表达式中的所有自变量。

在我们刚才的实例中，第一个归约，是兰姆达 x 这个较大的函数应用于 Sue 这个论元上，所以被替换的是 x，最后输出了一个关于 y 的函数，第二个归约，是兰姆达 y 这个较小的函数应用于 Sue 这个论元上，所以被替换的是 y，最后输出了一个关于 x 的函数。这些函数进一步应用于下一个论元上，才能输出一个具有真值的句子。

有的同学在这里可能会产生一个疑问，对于 love 这样的二元谓词，为什么不把它直接看作一个二元函数，即同时输入两个论元 x 和 y，为什么非要保持一元函数的设置呢？这涉及到了复杂的数学论证，这一过程叫做斯科林化，即在兰姆达演算系统中，所有函数都设置为一元的，即使是那些 n 元的函数也可以通过斯科林化转化为一元函数。

练习题：

1. 请大家用文字说明下列表达式分别表达了什么意思。

(a)λx∈ℕ.x>3 and x<7

(b)λx:x is a person.x loves formal semantics

(c)λX:X⊆D.[λy∈D.y∉X]

2. 请大家计算并化简下列式子。

(a)[λx.[λy.[λz.z introduced x to y]]](Ann)(Sue)

(b)[λx.[λy.[λz.z introduced x to y](Ann)](Sue)]

(c)[λx.[λy.[λz.z introduced x to y](Ann)]](Sue)

(d)[λx.[λy.[λz.z introduced x to y]](Ann)](Sue)

(e)[λf∈D<e,t>.[λx∈D.f(x)=1 and x is gray]] ([λy∈D.y is a cat])

(f)[λf∈D<e,<e,t»].[λx∈D.f(x)(Ann)=1]]([λy∈D.[λz∈D.z saw y]])

3. 求下列式子最终的语义类型

[λf∈D<e,<e,t».[λx∈D.f(x)(Ann)=1]]

5 组合性规则

组合性这一概念贯穿始终，现在我们已经学习了类型论以及非常基础的兰姆达演算，我们终于可以正式地引入组合性这一概念。我们在第一章节中讲到管约论的倒 Y 字模型，其中句法语义界面则以句法结构作为输入，最后经过一系列的规则，将句法结构或者更具体的说，是把生成出来的短语句法树转换为 LF 形式，我们把这一系列的规则称为组合性规则或者语义组合规则。

组合性规则（幼儿园版）：

Terminal Nodes(TN):

If α is a terminal node,then α is in the domain of $[\![]\!]$ if $[\![\alpha]\!]$ is in the lexicon.

对于终端节点，若这一节点的指称已经在词库中给定，那么我们就说这一节点在解释函数的定义域内，即可解释。

Non-branching Nodes(NN):

If α is a non-branching nodes,and β is its daughter node,then α is in the domain of $[\![]\!]$ if β is.In this case,$[\![\alpha]\!]=[\![\beta]\!]$.

对于句法树中的非分叉节点，如果其女儿节点（子节点）在解释函数的定义域内，那么该节点也在此定义域内，同时，非分叉母节点和女儿节点的指谓相同。

Functional Application(FA):

If α is a branching node and β,γ is the set of α's daughters,then α is

in the domain of $[\![]\!]$ if both β and γ are and $[\![\beta]\!]$ is a function whose domain contains $[\![\gamma]\!]$.In this case,$[\![\alpha]\!]$=$[\![\beta]\!]([\![\gamma]\!])$.

对于句法树中的分叉节点（严格二分叉），它有两个女儿节点，如果这两个女儿节点都在解释函数的定义域内，那么该节点也在解释函数的定义域内，同时其中一个女儿节点是一个函数，其定义域包含了另一个女儿节点的解释函数的定义域，那么我们就说，该母节点的指谓等于将前一个女儿节点应用在后一个女儿节点的输出结果。

Principle of Interpretability(可解释性原则):

All nodes in a phrase tree must be in the domain of the interpretation function $[\![]\!]$.

短语树中的所有节点都必须在解释函数的定义域内。

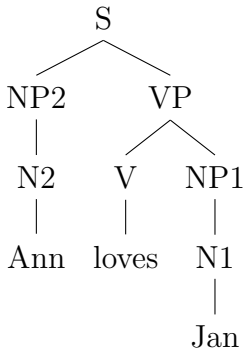
同时我们再回到蒙太古的观点，他认为句法的完备性和语义的可解释性是相吻合的，所有句法成分均需获得语义解释，同时语义的不可解释性也会导致句子不合法。例如：

*Ann laughed Jan.

这个句子为什么不合法，从句法角度来看因为这违反了题元准则，但是更本质的是这个句子在语义上是无法解读的，为什么呢，因为 laughed 是一个语义类型为 $\langle e,t\rangle$ 的动词，也就是说它只能被应用在语义类型为 e 的论元上，所以根据我们刚才讲到的组合规则，laughed 先要与 Jan 组合，因为 Jan 的语义类型是 e，所以输出了 t，但是 Ann 的语义类型是 e，因此 Ann 与 laughed Jan 无法进行组合，因为 e 和 t 无法进行组合，所以这个句子的语义是不可解释的。

下面我为大家演示如何将一个简单的短语句法树转换为 LF 表达式：

假设我们有如下句法树：



其语义推导如下：

⟦Jan⟧=Jan (根据组合性规则 TN)

⟦NP1⟧=⟦N1⟧=⟦Jan⟧=Jan (根据组合性原则 NN)

⟦loves⟧=[λy.[λz.z loves y]] (根据组合性规则 TN)

⟦V⟧=⟦loves⟧=[λy.[λz.z loves y]] (根据组合性规则 NN)

⟦VP⟧=⟦V⟧(⟦NP1⟧)=[λy.[λz.z loves y]](Jan)=[λz.z loves Jan] (根据组合性规则 FA)

⟦Ann⟧=Ann (根据组合性原则 TN)

⟦NP2⟧=⟦N2⟧=⟦Ann⟧=Ann (根据组合性原则 NN)

⟦S⟧=⟦VP⟧(⟦NP2⟧)=[λz.z loves Jan](Ann)=Ann loves Jan (根据组合性规则 FA)

练习：

请大家分别写出以上推导中每一步所对应的语义类型。

这里需要注意，有一些词并没有实际意义，例如 of、be、a、that 等，把它们叫做 semantically vacuous words，即语义空虚词。此外，对于某些非动词性的谓词，比如介词短语，例如“the city is in Texas”中，“in Texas”可以理解为将介词 in 这个函数应用到论元 Texas 上，即：

⟦in Texas⟧=⟦in⟧(⟦Texas⟧)=⟦in⟧(Texas)=[λx.[λy.y is in x]](Texas)=λy.y

is in Texas

所以这里的介词 in 的语义类型是 $\langle e, \langle e, t\rangle\rangle$，整个介词短语的语义类型是 $\langle e, t\rangle$。

我们还需要注意作为限制性修饰语（restrictive modifiers）的谓词（predicates），例如：

It is a city in Texas, 其中介词短语 PP 是一个限制性的修饰成分，如果按照我们之前的推算，PP 的语义类型是一个 $\langle e, t\rangle$，但是注意，a city 的语义类型也是 $\langle e, t\rangle$（我们前面讲过 a 是一个语义空虚成分），但是当我们想把它们两个组合起来时，才发现类型是不匹配的，因此无法进行组合。为了解决谓词限定性修饰成分的语义组合问题，我们引入了一个新的规则：

Predicate Modification(MP):

If α is a branching node,β,γ is the set of α's daughters, and $[\![\beta]\!]$ and $[\![\gamma]\!]$ are both in D<e,t>,then $[\![\alpha]\!]=\lambda x.[\![\beta]\!](x)=[\![\gamma]\!](x)=1$.

即：若两个姐妹节点都是类型为 $\langle e, t\rangle$ 的谓词，那么其母节点也是一个类型为 $\langle e, t\rangle$ 的谓词，其语义是这两个谓词的逻辑合取。

所以 $[\![\text{city in Texas}]\!]=\lambda x.[\![\text{city}]\!](x)=[\![\text{in Texas}]\!](x)=1$

$=\lambda x.$x is a city and x is in Texas

同时还需注意限定成分 the，从逻辑上讲，表限定的 the 有两层含义，第一层是存在性，即 $\exists x.f(x)$；第二层即唯一性，即至多存在一个 x，使得 f(x) 为真，或者我们可以这样说，如果有任意两个东西都满足 f，那么这两个东西必然是同一个东西，所以将这两个方面合并之后，我们就得到了 the 的语义解读，即：

$[\![\text{the}]\!]=\lambda P{:}\exists y[\forall x(P(x)\leftrightarrow x=y)].\tau x P(x)$

注意，其中 τ 是一个唯一性算子。我们可以看到 the 这个限定成分的语义类型是 $\langle\langle e, t\rangle, e\rangle$。即输入一个函数，返回一个实体 e。

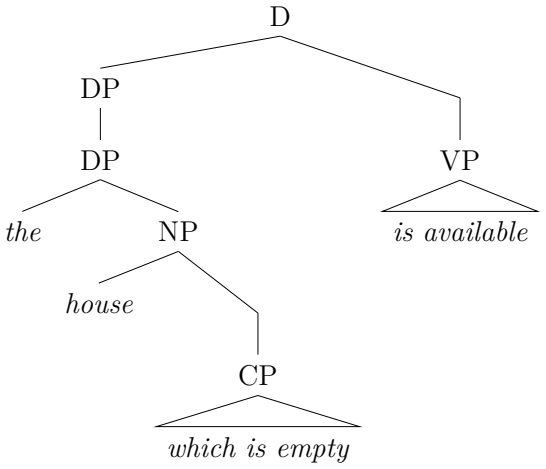
这个式子前半部分表示存在一个 y，使得对于任意 x，P(x) 为真，当且仅当 x=y, 即 P 只能被唯一的一个 y 满足；后半部分是一个唯一性算子，表示那个满足 P(x) 的唯一的 x。

6 基于变量的语义规则

前面的内容中，我们只是讨论了一些简单的结构，但是大家都知道，咱们的语言是复杂的，而且是递归的，在英语中，我们都学习过关系从句，那么我们不由发问，当关系从句作为谓词时，它的语义该如何解读呢？试看下面的句子：

The house which is empty is available.

很明显这是一个具有关系从句结构的句子，我们可以将其表达到句法树当中：



(这里向大家解释一下，因为大家还没有系统学习生成句法学中的技术

细节，因此我在此讲义中所绘制的句法树是非常粗糙的，大家不要以为这就是真正的句法结构，这里只是方便语义的组合计算，因此才给大家绘制了一个个不需要句法基础就能看明白的粗制滥造的句法树，不周不到之处，还请大家多多原谅)

那么对于 [CP which is empty] 的语义，我们应该如何解读呢？当然我们知道 empty 的语义解读：

⟦empty⟧=λx.x is empty

显然，其语义类型是 $\langle e,t\rangle$。那么整个从句的语义解读又是什么？

我们这里先采用奎因的猜想，他认为：

⟦which is empty⟧=⟦empty⟧

所以该 CP 的语义类型为 $\langle e,t\rangle$，而其修饰的成分 house 的语义类型也是一个 $\langle e,t\rangle$，根据我们之前得出的谓词修饰原则，house which is empty 的语义解读为：

⟦house which is empty⟧=λx.⟦house⟧(x)=⟦which is empty⟧(x)=1

=λx.x is a house and x is empty

所以 house which is empty 的语义类型是 $\langle e,t\rangle$。

然而，这只是基于奎因的假设，我们现在须证明这一假设，即 CP 是如何获得语义类型 $\langle e,t\rangle$ 的，以及根据句法学的理论，关系从句中涉及到移位，同时移位留下了语迹 t，那么语迹 t 的语义该如何解读呢？

这里我们引入了变量（variables）的概念，一个变量指称一个个体，同时既然它是变量，那么就必须要有给这一变量赋值的赋值函数（assignment），因此我们就定义了以下概念：

$[\![\alpha]\!]^a$:=the denotation of α under a, 其中 α 是一个句法树，而 a 则是一个赋值函数。

进而，我们可以说：

If α is a trace or pronoun,then ,for any assignment a,$[\![\alpha]\!]^a$=a.

因此我们这里引入了一个包含变量的语义规则，为了使我们的整套规则体系更具有兼容性，我们需要对现有的语义规则进行修订，修订如下：

组合性规则 (小学版)：

Terminal Nodes(TN):

If α is a terminal node occupied by a lexical item,then $[\![\alpha]\!]$ is specified in the lexicon and for any assignment $[\![\alpha]\!]$=$[\![\alpha]\!]^a$

Non-branching Nodes(NN):

If α is a non-branching nodes,and β is its daughter node,then for any assignment a,$[\![\alpha]\!]^a$=$[\![\beta]\!]^a$

Functional Application(FA):

If α is a branching node and β,γ is the set of α's daughters,then α is in the domain of $[\![]\!]$ if both β and γ are , for any assignment,$[\![\beta]\!]^a$ is a function whose domain contains $[\![\gamma]\!]^a$,then$[\![\alpha]\!]^a$ =$[\![\beta]\!]^a([\![\gamma]\!]^a)$.

Predicate Modification(PM)

If α is a branching node,β,γ is the set of α's daughters,then,for any assignment a,if $[\![\beta]\!]^a$ and $[\![\gamma]\!]^a$ are both functions of type<e,t>,then $[\![\alpha]\!]^a$=λx$\in$D.$[\![\beta]\!]^a$(x)=$[\![\gamma]\!]^a$(x)=1

修订了我们现有的语义组合规则之后，我们可以引入一个新的算法，即Predicate abstraction（谓词抽象）, 也叫兰姆达抽象：

If α is a branching node whose daughters are a relative pronoun and β,then $[\![\alpha]\!]$=λx.$[\![\beta]\!]^x$

那么我们该如何利用兰姆达抽象来推导语义式呢？试看下面的例句：

which John abandoned t.

我没有给出完整例句，这只是例句中的关系从句 CP，那么我该如何推导其语义呢，我来给大家演示一遍：

根据兰姆达抽象，可得：

$[\![\text{CP}]\!]=\lambda x.[\![\text{C'}]\!]^x$

再由空虚语义的理论，我们可以进一步化简：

原式 $=\lambda x.[\![\text{S}]\!]^x$

根据 FA，可得：

原式 $=\lambda x.[\![\text{VP}]\!]^x([\![\text{John}]\!]^x)$

根据 TN，可得：

原式 $=\lambda x.[\![\text{VP}]\!]^x([\![\text{John}]\!])$

根据 John 的自身属性，可得：

原式 $=\lambda x.[\![\text{VP}]\!]^x$(John) 根据 FA，可得：

原式 $=\lambda x.[\![\text{abandoned}]\!]^x([\![\text{t}]\!]^x)$(John)

根据语迹 t 的定义以及 TN，可得：

原式 $=\lambda x.[\![\text{abandoned}]\!]$(t)(John)

原式 $=\lambda x.[\lambda y.[\lambda z.z$ abandoned y]](x)(John)

原式 $=\lambda x.$John abandoned x

通过兰姆达抽象的法则，我们最终推导出了该关系从句的语义表达，也证实了奎因的猜想，即其语义类型为 <e,t>。

练习题：

请大家用兰姆达抽象的算法，推导下面这一句子中关系从句的语义：

(she is a girl)who sleeps.

但是这只是一个变量的情况，有很多句子中可以同时存在多个变量，例如下面的句子：

the man such that Mary reviewed the book which he wrote.

这是一个嵌套从句，最里面的句子 CP 其指示语位置上有一个 wh, 它与 wrote 后的语迹 t 同指，而 such 则与 he 同指。但是这只是我们掌握了

句法规则之后得出的，即我们在句法中引入了 co-indexing，但实际上在我们现存的语义组合规则中，是找不出来任意一条规则去解释这样的多变量现象。因此我们需要在现有的语义规则中引入变量的绑定或约束（binding）, 即我们可以对赋值函数进行重新定义：

A variable assignment is a partial function from $\mathbb{N}$ to D

即赋值函数是从自然数到实体集合的一个函数，也可以理解为把句法中的下标对应到具体的实体上。

那么修订后的 Traces and Pronouns Rule 就变成了：

If α is a pronoun or trace,a is a variable assignment, and i∈dom(a),then $[\![\alpha_i]\!]^a$=a(i).

例如，我们假设赋值函数 a 是：

$$\begin{bmatrix} 1 & Sue \\ 2 & Joe \end{bmatrix}$$

注意这是一个函数，是从自然数 1、2 到实体 Sue、Joe 的函数，表达的意思是若下标数字为 1，则返回 Sue，若下标数字为 2，则返回 Joe。其具体应用如下：

$[\![\text{he}_2]\!]^a$=a(2)=Joe

$[\![\text{t}_1]\!]^a$=a(1)=Sue

若赋值函数 a 是：

$$\begin{bmatrix} 1 & \text{"}Barriers\text{"} \\ 2 & Joe \end{bmatrix}$$

原句 α=he_2 wrote t_1, 试求 $[\![\alpha]\!]^a$

目前，为了解决多变量问题，我们引入了变量赋值函数，将句法中的下

标也纳入到了我们的规则系统中，但是我们只是给出了关于语迹和代词规则的修订，我们如何将这一变化引入我们的兰姆达抽象法则中呢？

在具体的修订之前，我们先定义一个 modified variable assignment:

Let a be an assignment,i∈ℕ,and x∈D.Then $[\![a]\!]^{x/i}$(a modified so as to assign x to i)is the unique assignment which fulfills the following conditions:

(i)dom($a^{x/i}$)=dom(a)∪i.

(ii)$a^{x/i}$(i)=x,and

(iii)for every j∈dom($a^{x/i}$) such that j≠i;$a^{x/i}$(j)=a(j)

我来解释以上三条，第一条是指修正之后的赋值函数的定义域要同时包括原赋值函数 a 的定义域以及下标 i 的集合，这就相当于在赋值函数中引入了下标，因此修正之后的赋值函数的定义域也须包含下标的集合。第二条是核心法则，即如何把 x 赋给 i。第三条的规定意思是，该函数只能改变一个值，其余值的指代是不随之发生变化的。

注意，我们规定对于 any tree，$[\![\alpha]\!]:=[\![\alpha]\!]^{\emptyset}$

我们现在可以修订我们的兰姆达抽象法则了：

If α is a branching node whose daughters are β_i and γ,where β is a relative pronoun or "such",and i∈ℕ,then for any variable assignment a,$[\![\alpha]\!]^a=\lambda x.[\![\gamma]\!]^{a^{x/i}}$.

请大家求 $[\![\text{such}_2 \text{ that Mary reviewed the book wh}_1 \text{ he}_2 \text{ wrote } t_1]\!]$

有关 variable binding 的内容这里省去，因为这一部分要求大家提前学习过约束理论，感兴趣的同学可阅读 Introduction to Government and Binding Theory(Liliane Haegeman) ，里面有详细的介绍。

7 广义量词

这里向大家致歉，之前发给大家的汇报安排是包含量词提升的，但是由于参与读书汇报的主要是本科生大一大二的同学，因此在此讲量词提升比较超纲，因为大家没有学习 Movement。因此我在手稿中删去了量词提升的内容，只留下了广义量词，希望大家海涵。

首先请大家辨析这里的量词，是指 every、some、no 等词，与我们汉语中的量词 classifier（个、只、头）有本质的区别。

其次，我们发问：量词的语义类型是什么？

最常见或者最直观的猜测是，量词的语义类型是类似于名词的 e 或者 $\langle e,t\rangle$，但是很明显不是，为什么呢？我们来论证。

对于 John came yesterday morning, 我们可以推出 John came yesterday.

但是来看一个包含量词的句子：

At most one letter came yesterday morning.

请问由这个句子，我们能推出 At most one letter came yesterday 吗？很显然不能。

再如，那些指称性的 DP（语义类型 e）在句法重组后真值条件是不发生改变的，例如：

I answered question 7.

Question 7,I answered.

John saw Mary.

Mary is such that John saw her.

John is such that Mary saw him.

但是考虑量化 DP 的例子：

Almost everybody answered at least one question.(几乎每个人都回答了至少一个问题)

At least one question,almost everybody answered.（至少存在一个问题，几乎所有人都回答了）

再比如：

It didn't snow on Christmas Day.

这个句子没有歧义。

但是：

It didn't snow on more than two of these days.

这个句子有两种解读，一是下雪的天数没有超过 2 天，二是超过 2 天没下雪。

所以综合以上考虑，我们人文量化 DP 的语义类型绝不等同于个体指称。

因此我们需要依靠我们现有的语义组合规则，反推出量化 DP 的语义类型。我们举一个简单的例子：

Every student read a book. 这一句中，read 是一个语义类型为 $\langle e, \langle e, t\rangle\rangle$ 的及物动词，a book 的语义类型为 e，那么动词短语 read book 的语义类型为 $\langle e, t\rangle$，这时 VP 需要进一步与量化 DP every student 组合，既然我们已经讲过量化 DP 不能是 e，显然在此处，也不能是 $\langle e, t\rangle$，因为如果是 $\langle e, t\rangle$ 的话，根据谓词修饰原则，整个句子得到的语义类型是 $\langle e, t\rangle$ 而不是 t，因此我们只能换一种角度思考，有没有一种可能，在这里，动词短语是一个论元，而量词短语是一个函数呢？在现有的条件下，我们只能这样思考，所以为了输出句子的语义类型 t，量词短语的语义类型只能是 $\langle\langle e, t\rangle, t\rangle$，即它作用于论元 VP 上，最后输出一个真值。请大家自行推出 every 的语义类型。

当然这里我们纯粹依靠了推理，那么推理出的结果有没有道理呢？当然有。量化 DP 和其他 DP 的核心区别在于量化 DP 表征的是一种关系，而不是指代，所以，量词的本质是两个集合之间的关系，第一个集合是量词量

化的对象，第二个集合是满足谓词性质的元素组成的集合。