

# **From Logic to Montague Grammar:**

## **Some Formal and Conceptual Foundations of Semantic Theory**

Course Handouts and Problem Sets

*Seth Cable  
University of Massachusetts Amherst*

January 13, 2014

## **Forward**

This document contains the class notes, handouts, and problem sets that were developed for the Fall 2013 Proseminar in Semantics at the University of Massachusetts Amherst.

At the suggestion of course mentor Barbara Partee, as well as several students in the class, I have collected these here and posted them publicly for anyone who may have an interest in either Montague Grammar (MG) generally or the specific papers “Universal Grammar” (UG) and “The Proper Treatment of Quantification in Ordinary English” (PTQ).

As to the content, structure, and goals of the course, I refer the reader to the course syllabus that follows this forward. I will state briefly, however, that the primary original contribution of these course notes to the already substantial didactic material on MG is its presentation of the algebraic framework of UG, its discussion of the relationship between the UG framework and the system presented in PTQ, and the introductory sections explaining and motivating the development of model-theoretic semantics.

Given that these are course notes and handouts, much of the material that follows borrows heavily from published texts and articles. A footnote at the beginning of each handout lists the readings that those notes are based upon (and often borrow from). As a service to the reader, however, I also list them comprehensively below:

### **Course Reading List**

#### Unit 1: Formal Preliminaries

- Heim & Kratzer. 1998. *Semantics in Generative Grammar*. Oxford: Dordrect.  
Partee, Barbara, Alice ter Meulen, and Robert E. Wall. 1993. *Mathematical Methods in Linguistics*. Dordrecht: Kluwer.  
Stewart, Ian and David Orme Tall. 1977. *The Foundations of Mathematics*. Oxford: Oxford University Press.

#### Unit 2: A Review of Propositional Logic and First Order Logic

- Crossley, J.N., C.J. Ash, C.J. Brickhill, J.C. Stillwell, and N.H. Williams. 1972. *What is Mathematical Logic?* New York: Dover.  
Gamut, L.T.F. 1991. *Logic, Language, and Meaning, Volume 1: Introduction to Logic*. Chicago: University of Chicago Press.  
Partee, Barbara, Alice ter Meulen, and Robert E. Wall. 1993. *Mathematical Methods in Linguistics*. Dordrecht: Kluwer.

#### Unit 3: Algebras and Semantics

- Dowty, David R., Robert E. Wall, and Stanley Peters (1981) *Introduction to Montague Semantics*. Dordrecht: Kluwer.  
Halvorsen, Per-Kristian and William Ladusaw. 1979. “Montague’s ‘Universal Grammar’: An Introduction for the Linguist.” *Linguistics and Philosophy* 3: 185-223.  
Partee, Barbara, Alice ter Meulen, and Robert E. Wall. 1993. *Mathematical Methods in Linguistics*. Dordrecht: Kluwer.  
Montague, Richard. (1974) “Universal Grammar.” In Thomason, Richmond (ed) *Formal Philosophy: Selected Papers of Richard Montague*. New Haven: Yale University Press.

#### Unit 4: Montague's Theory of Translation

- Dowty, David R., Robert E. Wall, and Stanley Peters (1981) *Introduction to Montague Semantics*. Dordrecht: Kluwer.
- Halvorsen, Per-Kristian and William Ladusaw. 1979. "Montague's 'Universal Grammar': An Introduction for the Linguist." *Linguistics and Philosophy* 3: 185-223.
- Montague, Richard. (1974) "Universal Grammar." In Thomason, Richmond (ed) *Formal Philosophy: Selected Papers of Richard Montague*. New Haven: Yale University Press.

#### Unit 5: An Algebraic Approach to Quantification and Lambda Abstraction

- Dowty, David R., Robert E. Wall, and Stanley Peters (1981) *Introduction to Montague Semantics*. Dordrecht: Kluwer.
- Partee, Barbara, Alice ter Meulen, and Robert E. Wall. 1993. *Mathematical Methods in Linguistics*. Dordrecht: Kluwer.
- Montague, Richard. (1974) "Universal Grammar." In Thomason, Richmond (ed) *Formal Philosophy: Selected Papers of Richard Montague*. New Haven: Yale University Press.

#### Unit 6: The Proper Treatment of Quantification in Ordinary English

- Dowty, David R., Robert E. Wall, and Stanley Peters (1981) *Introduction to Montague Semantics*. Dordrecht: Kluwer.
- Montague, Richard. (1974) "The Proper Treatment of Quantification in Ordinary English." In Thomason, Richmond (ed) *Formal Philosophy: Selected Papers of Richard Montague*. New Haven: Yale University Press.

I wish to thank here the students and visitors who participated in the seminar and whose comments and questions greatly improved the content of these notes: Elizabeth Bogal-Allbritten, Ed Ferrier, Cameron Gibbs, Hannah Greene, Hsin-Lun Huang, Dennis Kavlakoglu, Stefan Keine, Chisato Kitagawa, Jon Ander Mendia, Yangsook Park, Jeremy Pasquereau, Ethan Poole, and Megan Somerday.

Special thanks are owed first and foremost to our course mentor, Barbara Partee. Besides simply being an invaluable resource for us all, clarifying key points and drawing connections I would never have dreamed of, she was a constant font of encouragement both for the students and for myself. Without Barbara, this course never would have been possible.

Finally, if the reader notices any errors – either trivial or substantial – please do not hesitate to contact me.

## **Contents:**

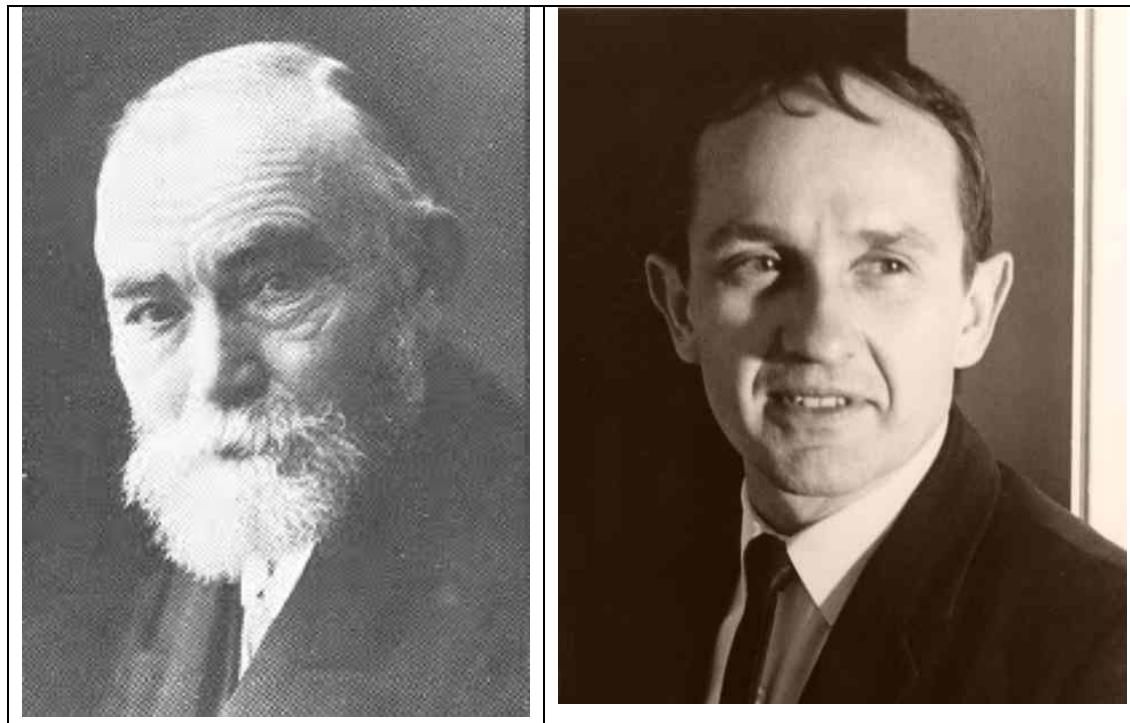
Given that the following material was originally separate documents, pagination begins afresh at the beginning of each section. Consequently, specific materials cannot be located by page number. However, I do provide here an overall outline of the material that follows.

0. Course Syllabus
1. Unit 1: Formal Preliminaries
  - 1.1 Relations and Functions
  - 1.2 Cardinalities, Infinities and Proof by Induction
  - 1.3 *Problem Set on the Formal Preliminaries*
2. Unit 2: A Review of Propositional Logic and First Order Logic
  - 2.1 Propositional Logic: Syntax and Natural Deduction
  - 2.2 First Order (Predicate) Logic: Syntax and Natural Deduction
  - 2.3 *Problem Set on Syntax and Natural Deduction*
  - 2.4 Propositional Logic: Formal Semantics and Valuations
  - 2.5 First Order Logic: Formal Semantics and Models
  - 2.6 *Problem Set on Formal Semantics (Valuations and Models)*
  - 2.7 Proving the Soundness and Completeness of Propositional Logic
  - 2.8 Key Applications of Model Theoretic Semantics for First Order Logic
3. Unit 3: Algebras and Semantics
  - 3.1 An Algebraic Perspective on Propositional Logic
  - 3.2 *Problem Set on Syntactic Operations, Semantic Operations, Homomorphisms*
  - 3.3 An Algebraic Perspective on the Syntax of FOL (Without Quantification)
  - 3.4 Montague's General Theory of Semantics
  - 3.5 *Problem Set on Languages and Interpretations*

4. Unit 4: Montague's Theory of Translation
  - 4.1 Laying the Groundwork
  - 4.2 The Notion of a 'Translation Base'
  - 4.3 Translation Bases and Interpretations
  - 4.4 *Problem Set on Translation and Indirect Interpretation*
5. Unit 5: An Algebraic Approach to Quantification and Lambda Abstraction
  - 5.1 Preliminaries
  - 5.2 Fregean Interpretations
  - 5.3 Computing with 'Logically Possible Partly-Fregean Interpretations'
  - 5.4 Applications to the Analysis of English
  - 5.5 *Problem Set on the Analysis of Quantification*
6. Unit 6: The Proper Treatment of Quantification in Ordinary English
  - 6.1 First Steps Towards PTQ: A New Presentation of Our System for Quantifiers
  - 6.2 The Fragment of English
  - 6.3 Intensional Logic
  - 6.4 The Translation System, Part 1
  - 6.5 The Translation System, Part 2
  - 6.6 *Problem Set on the PTQ System*

# **From Logic to Montague Grammar: Some Formal and Conceptual Foundations of Semantic Theory**

## **Syllabus**



Linguistics 720  
Tuesday, Thursday 2:30 – 3:45  
Room: Dickinson 110  
Course Instructor: Seth Cable  
Course Mentor: Barbara Partee

Course Website:  
<http://people.umass.edu/scable/LING720-FA13/>  
E-mail: [scable@linguist.umass.edu](mailto:scable@linguist.umass.edu)  
E-mail: [partee@linguist.umass.edu](mailto:partee@linguist.umass.edu)

### **1. General Overview**

In general, the semantics proseminar is intended to serve as a bridge between the introductory graduate semantics courses (610, 620) and the more advanced semantics seminars. Typically, a specific subject is covered in more depth than is normally done in the intro classes, but the discussion is paced at a level appropriate for second- and third-year students.

The subject of this year's proseminar is, loosely speaking, the formal semantic framework of Montague Grammar. The overall goal of the course will be to provide students with enough formal background to successfully navigate and critically evaluate the early literature of our discipline. Consequently, this course will be closer in spirit to an introductory semantics course than to a full-fledged seminar. It is my hope that – with input from the students and other participants – this course may ultimately be developed into a regularly taught component of our graduate semantics curriculum.

## **1.1 Goals of the Course**

The primary goal of this course is to provide students who have taken 610 and 620 with (some of) the background necessary to read papers written within the Montague Grammar tradition, especially the classic papers in Montague (1974) and Partee (1976).

As the field of formal semantics has grown over the past forty years, it has become more tightly integrated into the other subdisciplines of linguistics – especially syntax. A consequence of this integration is that much semantic research nowadays assumes a GB-style syntactic architecture, where LF tree-structures are separately generated by the syntax and then ‘input’ to a recursively defined semantic interpretation operation. In addition, in much current work, the denotations output by the interpretation operation are characterized purely via representations in a logical metalanguage, one that is implicitly understood by readers.

Although there are many merits to these developments, a negative consequence is that one can be expert in reading and evaluating current semantic research, while nevertheless finding great difficulty in comprehending the earliest and most influential works of our discipline. In addition, although the original ‘Montague Grammar’ framework is no longer widely used, much current work is nevertheless still written in an explicitly ‘model-theoretic’ style akin to those early works. Finally, the research done within frameworks that are ‘directly compositional’ is best understood and contextualized as a development of the key ideas of Montague.

For these reasons, it is quite important for semantics students to have the ability to read and comprehend the classic papers of Montague (1974) and Partee (1976). It is also extremely important to be able to ‘translate’ proposals from one research tradition into another, to allow for effective comparison and evaluation of different analyses. The primary goal of this seminar, then, is to develop these skills, alongside a deeper understanding and appreciation of the seminal works of Montague.

## **1.2 Structure of the Course**

A fundamental concept in the work of Montague and others is that of ‘interpretation with respect to a model’. While this concept is emphasized in some semantics textbooks (Chierchia & McConnell-Ginet 2000), it is not emphasized in all introductory semantics curricula. In addition, in many semantics textbooks, the concept is introduced somewhat by fiat, without much motivating (or clarifying) context. For this reason, our course will begin by providing some crucial historical and conceptual context for the tools of ‘model-theoretic semantics’.

In this first section, “Why Models?”, we begin by reviewing the syntax and proof system of two fundamentally important logical languages: Propositional Logic (PL) and First Order Logic (FOL). We will then see how several key questions about these systems motivate the development of a mathematically precise characterization of what it means to be an ‘interpretation’ of these languages. We will then see how so-called ‘models’ can play this role for FOL. Finally, we will see how, once armed with the notion of a ‘model’, we can answer those fundamentally important questions about FOL. This general plot structure is outlined below:

## Part 1: Why Models?

- Propositional Logic (PL): Syntax and Natural Deduction System
- First Order (Predicate) Logic (FOL): Syntax and Natural Deduction System
- Formal Semantics of PL: Valuations
- Proving Soundness and Completeness of PL Natural Deduction
- Formal Semantics of FOL: Models
- Proof Sketch of Soundness and Completeness of FOL Natural Deduction
- Some Other Neat Results of Model Theory
- A Model Theory for *Natural Language*? Benefits and Obstacles

### Readings and Resources (Posted on Moodle)

- Gamut, L.T.F. 1991. *Logic, Language, and Meaning, Volume 1: Introduction to Logic*. Chicago: University of Chicago Press. (Chapters 1, 2, 3, 4)
- Partee, Barbara, Alice ter Meulen, and Robert E. Wall. 1993. *Mathematical Methods in Linguistics*. Dordrecht: Kluwer. (Chapters 5, 6, 7)
- Crossley, J.N., C.J. Ash, C.J. Brickhill, J.C. Stillwell, and N.H. Williams. 1972. *What is Mathematical Logic?* New York: Dover. (Chapter 2)

As outlined above, once it's clear what can be achieved with a model theoretic semantics for FOL, the question will naturally arise of whether such a semantics could ever be given for a *natural language*. We'll examine the obvious benefits of developing such a semantic theory, as well as some of the obvious obstacles that initially stood in its way.

Next, the class will veer sharply into the domain of abstract algebra. Many of the key ideas in Montague's work are based on the following core insight: interpreting a language with respect to a model can be conceived of mathematically as a special kind of 'mapping' between algebras, namely a 'homomorphism'. Therefore, we will begin with an introduction to the concepts of an 'algebra' and a 'homomorphism'. We'll then see right away that a 'valuation' of PL is essentially a homomorphism from a kind of 'syntactic algebra' (forming the sentences of PL) to a 'semantic algebra' (consisting of operations over truth-values). We'll then further develop this notion of 'interpretation as homomorphism' so that it can apply to FOL (without quantification).

## Part 2: Algebras and Semantics

- Introduction to Algebras and Morphisms.
- PL as an Algebra, Valuations as Morphisms
- Algebraic Syntax/semantics of FOL (without quantification)

### Readings and Resources (Posted on Moodle)

- Partee, Barbara, Alice ter Meulen, and Robert E. Wall. 1993. *Mathematical Methods in Linguistics*. Dordrecht: Kluwer. (Chapter 9)
- Halvorsen, Per-Kristian and William Ladusaw. 1979. "Montague's 'Universal Grammar': An Introduction for the Linguist." *Linguistics and Philosophy* 3: 185-223.
- Dowty, David R., Robert E. Wall, and Stanley Peters (1981) *Introduction to Montague Semantics*. Dordrecht: Kluwer (Chapter 8)

At this point, we will have enough background to begin discussing Montague's theory of translation. Another key part of Montague's (1974) framework is the notion that – under certain, very special conditions – translation from one language into another can also be conceived of as a homomorphism between (syntactic) algebras. We'll see that this 'homomorphic' conception of translation has a very crucial consequence: *if a language L can be (homomorphically) translated into another language L', which has a defined (model-theoretic) semantics, then you've also thereby provided L with a defined (model-theoretic) semantics.* We will discuss the central importance of this consequence for the general program of formal semantics.

### Part 3: Algebras, Translations, and Indirect Interpretation

- Montague's Theory of Indirect Interpretation
- Indirect Interpretation of English (without quantification)
- Algebraic Syntax and Semantics of FOL (with quantification)
- Indirect Interpretation of Quantification in English

#### Readings and Resources (Posted on Moodle)

Halvorsen, Per-Kristian and William Ladusaw. 1979. "Montague's 'Universal Grammar': An Introduction for the Linguist." *Linguistics and Philosophy* 3: 185-223.

Dowty, David R., Robert E. Wall, and Stanley Peters (1981) *Introduction to Montague Semantics*. Dordrecht: Kluwer (Chapter 8)

Montague, Richard. (1974) "Universal Grammar." In Thomason, Richmond (ed) *Formal Philosophy: Selected Papers of Richard Montague*. New Haven: Yale University Press.

Once we've reached this point, we'll actually have developed a significant purely 'extensional' fragment of English in Montague's (1974) framework. However, many of the key advances in Montague's work stem from his use of a specially designed 'Intensional Logic' (IL). Thus, the final stage of our gradual introduction to Montague's system will be a study of his Intensional Logic, as well as some its basic applications to the analysis of English.

### Part 4: Montague's Intensional Logic (IL) and its Applications

- Key concepts behind the Intensional Logic (IL)
- Formal syntax and model-theoretic semantics of IL
- Algebraic characterization of IL's model-theoretic semantics
- Indirect Interpretation of English via IL (Basic Examples)

#### Readings and Resources (Posted on Moodle)

Dowty, David R., Robert E. Wall, and Stanley Peters (1981) *Introduction to Montague Semantics*. Dordrecht: Kluwer (Chapter 6, Chapter 7)

Gamut, L.T.F. 1991. *Logic, Language, and Meaning, Volume 2: Intensional Logic and Logical Grammar*. Chicago: University of Chicago Press. (Chapter 5)

Montague, Richard. (1974) "Universal Grammar." In Thomason, Richmond (ed) *Formal Philosophy: Selected Papers of Richard Montague*. New Haven: Yale University Press.

Having come this far, it will now be possible for us to read and discuss much of Montague's seminal paper "Universal Grammar" (UG). Our guide to this work will be the excellent overviews by Halvorsen & Ladusaw (1979) and Dowty *et al.* (1981). Having walked through the most important parts of UG, we will also be well-equipped to read and discuss that most seminal of Montague's works, "The Proper Treatment of Quantification in Ordinary English" (PTQ). If time permits, we will also read one or two other classic works in formal semantics, such as Karttunen's "Syntax and Semantics of Questions".

### Part 5: Classic Papers in Montague Grammar

- Montague's "Universal Grammar"
- Montague's "The Proper Treatment of Quantification in Ordinary English"
- (Perhaps Karttunen's "Syntax and Semantics of Questions")
- (Perhaps one more paper, from Partee (1976))

### Readings and Resources (Posted on Moodle)

Halvorsen, Per-Kristian and William Ladusaw. 1979. "Montague's 'Universal Grammar': An Introduction for the Linguist." *Linguistics and Philosophy* 3: 185-223.

Dowty, David R., Robert E. Wall, and Stanley Peters (1981) *Introduction to Montague Semantics*. Dordrecht: Kluwer (Chapter 8)

Montague, Richard. (1974) "Universal Grammar." In Thomason, Richmond (ed) *Formal Philosophy: Selected Papers of Richard Montague*. New Haven: Yale University Press.

Montague, Richard. (1974) "The Proper Treatment of Quantification in Ordinary English" In Thomason, Richmond (ed) *Formal Philosophy: Selected Papers of Richard Montague*. New Haven: Yale University Press.

Dowty, David R., Robert E. Wall, and Stanley Peters (1981) *Introduction to Montague Semantics*. Dordrecht: Kluwer (Chapter 7)

Gamut, L.T.F. 1991. *Logic, Language, and Meaning, Volume 2: Intensional Logic and Logical Grammar*. Chicago: University of Chicago Press. (Chapter 6)

Partee, Barbara. 1975. "Montague Grammar and Transformational Grammar." *Linguistic Inquiry* 6(2): 203-300

Partee, Barbara. 1976. *Montague Grammar*. New York: Academic Press

### Some Additional Notes About Course Content and Difficulty

- *This is not an introduction to semantics.* I assume students have taken (the equivalent) of 610 and 620. Thus, many technical concepts from 610 and 620 will be assumed. (e.g., extension, intension, possible world semantics, etc.)
- *This is not an introduction to logic.* As explained in the course announcement, I assume that students have a basic background in logic, as would be obtained in a typical undergraduate introduction to logic course. (e.g. 'translation' into PL and FOL, truth-tables, natural deduction)
- ***Doing the reading is critical.*** I'm going to aim to move relatively quickly, particularly through Part 1. This course is cumulative, and so it's crucial that you keep up.

### **General Piece of Advice: MEET WITH US!**

At any point in the semester, please meet with Barbara and I regarding any issues at all, particularly if you are having any kind of difficulties with the course. We are also very happy to discuss anything at all, especially any interesting puzzles you happen to note along the way.

## **2. Course Requirements**

As mentioned, this class will be closer in spirit to 620 than to a full-fledge seminar. Thus, the course requirements will be similar to those of 620.

### **2.1 Problem Sets**

Due to the technical nature of the course material, it is critical that students complete regular weekly problems sets. These problem sets will be assigned on Thursday and due the following Thursday. Students are permitted to collaborate on problem sets, as long as each student writes up their work individually. The answers to the problem sets will not always be discussed in class. In such cases, I will write up and distribute answer keys.

### **2.2 Final Presentation and Final Paper**

Students will be required to complete a final project, which will be presented in two formats: an in-class presentation at the end of the term, and a final paper.

The nature of this final project is rather open-ended, but any of the following types of project would be acceptable:

- A critical discussion of some aspect of Montague's classic papers UG or PTQ which we did not cover in class.
- A critical discussion of some paper in Partee (1976) (not covered in class)
- A critical discussion of some paper in Barker & Jacobson (2007) *Direct Compositionality*
- A Montague Grammar (MG) or Directly Compositional (DC) adaptation of an existing semantic analysis
- An original analysis of some phenomenon, done within MG or DC

The in-class presentations will take place during a special meeting *after* the final class; ideally during the week of December 9<sup>th</sup> – December 13<sup>th</sup>. The final paper will be due December 20<sup>th</sup>.

**Students should decide upon a final project by November 14<sup>th</sup> (at the absolute latest).**

### **3. Various Dates of Interest**

October 15<sup>th</sup>: No class (Monday schedule)

**November 14<sup>th</sup>:** **Declare topic of final project**

November 28<sup>th</sup>: No class (Thanksgiving break)

December 5<sup>th</sup>: Last day of class

**December 9<sup>th</sup>-13<sup>th</sup>:** **Final Presentations**

**December 20<sup>th</sup>:** **Final Paper Due**

**December 23<sup>rd</sup>:** **Final Grades Due**

## **Unit 1:**

### **Formal Preliminaries**

## Formal Preliminaries, Part 1: Relations and Functions<sup>1</sup>

### 1. Basic Concepts of Set Theory

I assume that the reader is familiar with the following:

- Basic concept of a set
- Set abstraction notation, { x : x is a boy }
- The concepts of cardinality, subset, powerset, union, intersection, complementation
- Various key set theoretic identities ( $X \cup (Y \cap Z) = ((X \cup Y) \cap (X \cup Z))$ )

For a review of these key concepts, the reader is referred to Partee *et al.* (1993), Chapter 1

### 2. Relations and Functions

#### 2.1 Ordered Tuples, Products, Projections

##### (1) Ordered Pair

The defining property of an ordered pair is that ‘order matters’. That is:

- If  $\langle x, y \rangle = \langle y, x \rangle$ , then  $x = y$
- If  $x \neq y$ , then  $\langle x, y \rangle \neq \langle y, x \rangle$

##### (2) Ordered $n$ -Tuple

With the notion of an ordered pair, we can define the concept of an ordered triple:

$$\langle x, y, z \rangle =_{def} \langle \langle x, y \rangle, z \rangle$$

Clearly, this will generalize to the definition of an arbitrary  $n$ -tuple:

$$\langle x_1, \dots, x_n \rangle =_{def} \langle \langle x_1, \dots, x_{n-1} \rangle, x_n \rangle$$

##### (3) Cartesian Product

If we have two sets A, B, then  $A \times B$ , the *cartesian product of A and B* is:

$$A \times B =_{def} \{ \langle x, y \rangle : x \in A \text{ and } y \in B \}$$

Illustration:     $\{ a, b \} \times \{ c, d \} = \{ \langle a, c \rangle, \langle a, d \rangle, \langle b, c \rangle, \langle b, d \rangle \}$   
                       $\{ c, d, e \} \times \{ a, b \} = \{ \langle c, a \rangle, \langle c, b \rangle, \langle d, a \rangle, \langle d, b \rangle, \langle e, a \rangle, \langle e, b \rangle \}$

Note: For any set A,  $A \times \emptyset = \emptyset$

---

<sup>1</sup> These notes are based upon material in the following required readings: Partee *et al.* (1993) Chapter 1, Chapter 2; Heim & Kratzer (1998) pp. 29-32; Stewart & Tall (1977) pp. 105-106.

(4) **n-Ary Cartesian Product**

Suppose that  $A_1, \dots, A_n$  is a series of n sets:

$$A_1 \times \dots \times A_n =_{\text{def}} \text{the set of all } n\text{-tuples } \langle a_1, \dots, a_n \rangle \text{ such that } a_1 \in A_1, \dots, a_n \in A_n \\ =_{\text{def}} \{ \langle a_1, \dots, a_n \rangle : a_i \in A_i \}$$

Illustration:

$$\{ a, b \} \times \{ c, d \} \times \{ e, f \} = \{ \langle ace \rangle, \langle acf \rangle, \langle ade \rangle, \langle adf \rangle, \langle bce \rangle, \langle bcf \rangle, \langle bde \rangle, \langle bdf \rangle \}$$

(5) **Cartesian Power**

- Let  $A$  be any set.  $A^2 = A \times A$
  - Let  $A$  be any set and  $n$  be any (natural) number.  $A^n = A \times \dots \times A$  ( $n$  times)
- 

## 2.2 Relations, Domains, Ranges, Inverses

(6) **(Binary) Relation**

A set of ordered pairs is a *(binary) relation*.

Illustrations:  $\{ \langle x, y \rangle : x \text{ is the mother of } y \}$   
 $\{ \langle x, y \rangle : x \text{ loves } y \}$   
 $\{ \langle a, b \rangle, \langle c, d \rangle, \langle e, f \rangle, \dots \}$

(7) **Domain and Range**

Let  $R$  be a relation.

- a. The Domain of  $R$ :  $\{ x : \langle x, y \rangle \in R \}$
- b. The Range of  $R$ :  $\{ y : \langle x, y \rangle \in R \}$

(8) **Inverse of Relation**

Let  $R$  be a relation.  $R^{-1}$  is the *inverse of  $R$* , and is defined as follows:

$$R^{-1} =_{\text{def}} \{ \langle y, x \rangle : \langle x, y \rangle \in R \}$$

Illustration: Let  $R = \{ \langle a, b \rangle, \langle c, d \rangle, \langle e, f \rangle \}$   
 $R^{-1} = \{ \langle b, a \rangle, \langle d, c \rangle, \langle f, e \rangle \}$

Note:  $(R^{-1})^{-1} = R$

(9) **n-Ary Relation**

An  $n$ -ary relation  $R$  is a set of  $n$ -tuples.

Illustrations:  $\{ \langle x, y, z \rangle : x \text{ is between } y \text{ and } z \}$  ternary relation  
 $\{ \langle x, y, z, s \rangle : x \text{ bet } y \text{ } z \text{ dollars that } s \text{ would lose } \}$  quaternary relation  
 $\{ \langle a, b, c, d, e \rangle, \langle f, g, h, i, j \rangle, \langle k, l, m, n, o \rangle \}$  ‘5-ary’ relation

Note: Given the definition in (2), an  $n$ -ary relation  $R$  is equivalent to a binary relation whose domain is a set of  $(n-1)$ -tuples, (i.e., an  $(n-1)$ -ary relation).

## 2.3 Functions and Stuff

### (10) Function

Let  $R \subseteq A \times B$ .  $R$  is a *function from A to B* if the following conditions hold:

- a. The domain of  $R$  is  $A$
- b. If  $\langle x, y \rangle \in R$  and  $\langle x, z \rangle \in R$ , then  $y = z$ .

### (11) Key Notations Relating to Functions

- a.  $f: A \rightarrow B \quad =_{def} \quad f$  is a function from  $A$  to  $B$
- b.  $f(x) \quad =_{def} \quad$  the unique  $y$  s.t.  $\langle x, y \rangle \in f$
- c.  $B^A \quad =_{def} \quad$  the set of all functions from  $A$  to  $B$ .
- d.  $f: A \rightarrow B \rightarrow C \quad =_{def} \quad f$  is a function from  $A$  to  $C^B$

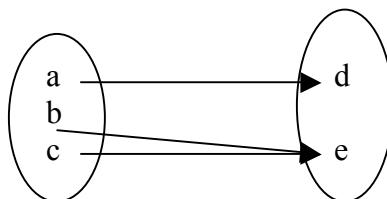
### (12) Surjection, Injection, Bijection

Let  $f: A \rightarrow B$ .

- a.  $f$  is a surjection (onto)  $=_{def}$  the range of  $f = B$   
 $=_{def}$  for every  $b \in B$ , there is an  $a \in A$  s.t.  $f(a) = b$
- b.  $f$  is an injection (one-to-one)  $=_{def}$  if  $f(a) = f(a')$ , then  $a = a'$   
 $=_{def}$  each  $a \in A$  is mapped to a different  $b \in B$
- c.  $f$  is a bijection  $=_{def}$   $f$  is a surjection (onto) and an injection (one-to-one)

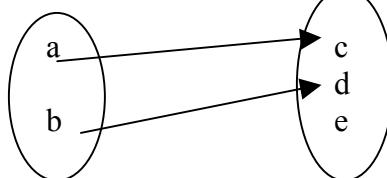
Illustrations:

Surjection:



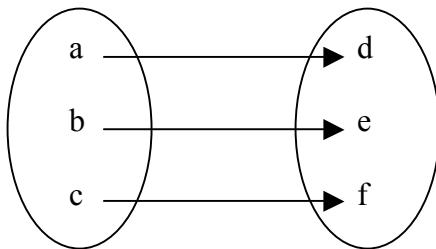
(not an injection)

Injection:



(not a surjection)

Bijection:



(13) **Inverses and Bijections**

- a. Since a function  $f$  is a relation, we can speak of its inverse  $f^{-1}$
- b. The inverse of a function  $f$  isn't *necessarily* a function (consider surjections)
- c. **If  $f: A \rightarrow B$  is a bijection, then  $f^{-1}$  is a function.  $f^{-1}$  is also a bijection.**

Proof:

(i)  $f^{-1}$  is a function.

Suppose it weren't. Then there would be  $\langle x, y \rangle, \langle x, z \rangle \in f^{-1}$  where  $y \neq z$ . But this would entail  $f(z) = f(y) = x$ , and so  $f$  isn't an injection, and so  $f$  isn't a bijection, contrary to assumption.

(ii)  $f^{-1}$  is an injection (one-to-one).

Suppose it weren't. Then  $\langle x, y \rangle, \langle z, y \rangle \in f^{-1}$  where  $x \neq z$ . But this would entail that  $\langle y, x \rangle, \langle y, z \rangle \in f$ , where  $x \neq z$ , and so  $f$  isn't a function, contrary to assumption.

(iii)  $f^{-1}$  is a surjection (onto).

Suppose it weren't. Then there is an  $x \in A$  such that there is no  $y \in B$  such that  $f^{-1}(y) = x$ . But, then it follows that that there is an  $x \in A$  such that there is no  $y \in B$  such that  $f(x) = y$ , and so the domain of  $f$  isn't  $A$ , contrary to assumption.

(14) **n-Ary Function**

- Note that the domain of a function can be a set of  $n$ -tuples. Such a function will be dubbed an ' $n$ -ary function'.

$$f: (A_1 \times \dots \times A_n) \rightarrow B$$

Illustration:     $\{ \langle \langle x, y \rangle, z \rangle : z = x + y \}$       binary function  
 $\{ \langle \langle x, y, z \rangle, s \rangle : s = x + y + z \}$       ternary function

- Note that an  $n$ -ary function is a set of ordered pairs, the first member of which is an  $n$ -tuple.
- **Consequently, given the definition in (2), an  $n$ -ary function is equivalent to an  $(n+1)$ -ary relation**

Illustration:

$$\begin{array}{ccc} \{ \langle \langle x, y \rangle, z \rangle : z = x + y \} & = & \{ \langle x, y, z \rangle : z = x + y \} \\ \{ \langle \langle x, y, z \rangle, s \rangle : s = x + y + z \} & = & \{ \langle x, y, z, s \rangle : s = x + y + z \} \end{array}$$

Note: Sometimes the term 'function' is restricted to *unary* functions. Functions of arity greater than 1 (*i.e.*, binary, ternary, etc.) are sometimes dubbed 'operations'.

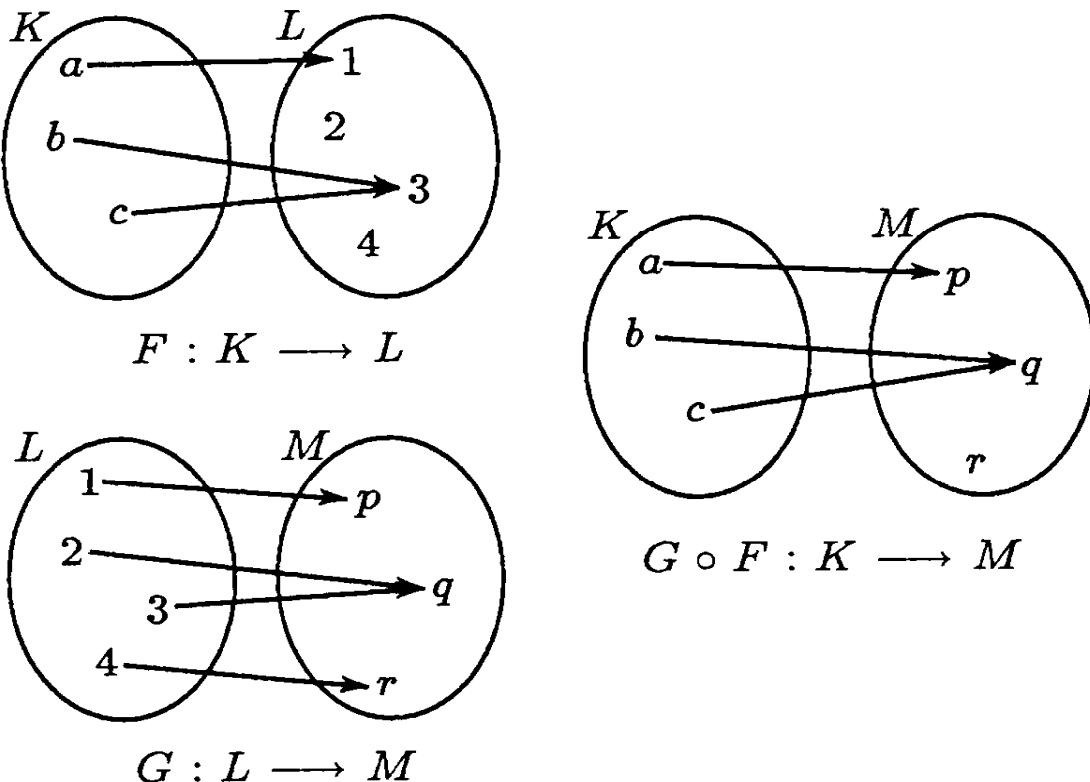
## 2.4 Function Composition

### (15) (Basic) Function Composition

Let  $f: A \rightarrow B$  and  $g: B \rightarrow C$  be unary functions. The *composition of f and g* is

$$g \circ f = \{ \langle x, z \rangle : \text{for some } y, \langle x, y \rangle \in f \text{ and } \langle y, z \rangle \in g \}$$

Illustration: From Partee et al. (1993):



**Figure 2–3: Composition of two functions  $F$  and  $G$**

Note: For all  $x$ ,  $g \circ f(x) = g(f(x))$

Note: Another way of writing ‘the composition of  $f$  and  $g$ ’ is  $g \circ f$

Note: In the definitions above, the functions  $g$  and  $f$  are assumed to be unary...  
We can generalize these definitions to functions of any arity...

### (16) (Generalized) Function Composition

Let  $g$  be an  $n$ -ary function, and let  $f_1, \dots, f_n$  be a series of  $n$   $m$ -ary functions. The *composition of  $g$  and  $f_1, \dots, f_n$*  is the  $m$ -ary function defined as follows:

$$\begin{aligned} g\langle f_1, \dots, f_n \rangle &=_{def} \text{the } m\text{-ary function such that for any } m\text{-ary sequence } a_1, \dots, a_m \\ &\quad g\langle f_1, \dots, f_n \rangle(\langle a_1, \dots, a_m \rangle) = \\ &\quad g(f_1(\langle a_1, \dots, a_m \rangle), \dots, f_n(\langle a_1, \dots, a_m \rangle)) \end{aligned}$$

Illustration:

Let  $g = \{ \langle x, y \rangle, z : z = x + y \}$ ,  $f = \{ \langle x, y \rangle : y = x - 1 \}$ ,  $h = \{ \langle x, y \rangle : y = x + 2 \}$

$$\text{Then: } g\langle f, h \rangle(2) = g(f(2), h(2)) = g(1, 4) = 5$$

$$g\langle f, h \rangle = \{ \langle x, y \rangle : y = (x-1) + (x+2) \}$$


---

## 3. Characteristic Functions

### (17) Characteristic Functions and Characteristic Sets

#### a. Characteristic Function $f_A$ of a Set A

For every set  $A \subseteq B$ , there is a unique function  $f_A: B \rightarrow \{1, 0\}$ , defined below:

$$\text{For all } b \in B, f_A(b) = 1 \text{ iff } b \in A$$

#### b. Characteristic Set $A_f$ of a Function $f: B \rightarrow \{1, 0\}$

For every function  $f: B \rightarrow \{1, 0\}$ , there is a unique set  $A_f \subseteq B$ , defined below:

$$\text{For all } b \in B, b \in A_f \text{ iff } f(b) = 1$$

Note:

- Given this regular correspondence between sets and (characteristic) functions, we will often shift freely between the two without comment.
- Note, however, that such equivocation is not entirely innocent: sets are distinct set-theoretic objects from their characteristic functions.*

$$\{a, b, c\} \neq \{ \langle a, 1 \rangle, \langle b, 1 \rangle, \langle c, 1 \rangle, \langle d, 0 \rangle, \langle e, 0 \rangle, \langle f, 0 \rangle \}$$

### (18) Characteristic Functions of Relations

Recalling the definition in (9) of an  $n$ -ary relation, the definition in (17) entails that every  $n$ -ary relation  $R \subseteq A_1 \times \dots \times A_n$  has its own characteristic function:

$$\text{For all } \langle a_1, \dots, a_n \rangle \in A_1 \times \dots \times A_n, f_R(\langle a_1, \dots, a_n \rangle) = 1 \text{ iff } \langle a_1, \dots, a_n \rangle \in R$$

Note: For this reason, we will also often shift freely between  $n$ -ary relations and  $n$ -ary characteristic functions (even though they are, strictly speaking, different objects)

(19) **One Final Observation**

- The characteristic function of an n-ary relation is an n-ary function.
- But, recall from (14), that every n-ary function is equivalent to an  $(n+1)$ -ary relation
- **Thus, every n-ary relation corresponds to a unique  $(n+1)$ -ary relation**

Illustration:  $\{ \langle a,b \rangle, \langle b,c \rangle, \langle c,d \rangle \} \approx \{ \langle a,b,1 \rangle, \langle b,c,1 \rangle, \langle c,d,1 \rangle, \langle b,a,0 \rangle, \langle c,b,0 \rangle, \langle d,c,0 \rangle, \dots \}$

Note: Sometimes, Montague shifts freely between these two objects (so, watch out!)

---

4. **Currying Functions**<sup>2</sup>

(20) **Currying of Binary Functions**

Let  $f$  be a binary function  $f: (A \times B) \rightarrow C$ . There's a unique function  $curry(f): A \rightarrow B \rightarrow C$  defined as follows:

$$\text{For all } \langle x,y \rangle \in A \times B, \quad f(\langle x,y \rangle) = c \quad \text{iff} \quad curry(f)(x)(y) = c$$

Illustration:

Suppose we have the following function  $f: \{ a, b \} \times \{ c, d \} \rightarrow \{ 1, 0 \}$

$$f: \begin{pmatrix} \langle a,c \rangle \rightarrow 1 \\ \langle a,d \rangle \rightarrow 0 \\ \langle b,c \rangle \rightarrow 0 \\ \langle b,d \rangle \rightarrow 1 \end{pmatrix} \quad curry(f): \begin{pmatrix} a \rightarrow \begin{pmatrix} c \rightarrow 1 \\ d \rightarrow 0 \end{pmatrix} \\ b \rightarrow \begin{pmatrix} c \rightarrow 0 \\ d \rightarrow 1 \end{pmatrix} \end{pmatrix}$$

Note: The lambda notation makes it quite easy to define  $curry(f)$ :

$$curry(f) = [ \lambda x : [ \lambda y : f(\langle x,y \rangle) ] ]$$

---

<sup>2</sup> Given that the original discoverer of this technique was Moses Schönfinkel, some have proposed that the term ‘schönfinkeling’ be used instead of ‘currying’ (as Haskell Curry rediscovered the technique later). However, to my knowledge, the term ‘schönfinkeling’ has not widely caught on in mathematics and computer science.

## (21) Currying n-Ary Functions

- Note that in the definition in (20), the set A could itself be a set of pairs.
  - In such a case,  $f(<< x, y >, z >) = c \text{ iff } \text{curry}(f)(< x, y >)(z) = c$
- Thus, in such a case,  $\text{curry}(f)$  will itself be a binary function. Thus, we could easily also speak of the function  $\text{curry}(\text{curry}(f))$ .
  - In such a case  $f(<< x, y >, z >) = c \text{ iff } \text{curry}(\text{curry}(f))(x)(y)(z) = c$
  - This reasoning clearly generalizes to the following:

Let  $f$  be an  $n$ -ary function  $f: A_1 \times \dots \times A_n \rightarrow C$ . There is a unique function  $\text{CUR}(f)$ ,  $\text{CUR}(f): A_1 \rightarrow \dots \rightarrow A_n \rightarrow C$  defined as follows:

$$\begin{aligned} \text{For all } & < a_1, \dots, a_n > \in A_1 \times \dots \times A_n, \\ & f(< a_1, \dots, a_n >) = c \quad \text{iff} \quad \text{CUR}(f)(a_1) \dots (a_n) = c \end{aligned}$$

Note: The lambda notation again makes it quite easy to define  $\text{CUR}(f)$

$$\text{CUR}(f) = [ \lambda x_1 : \dots [ \lambda x_n : f(< x_1, \dots, x_n >) ] \dots ]$$

## (22) Key Consequence

Let  $R \subseteq A_1 \times \dots \times A_n$  be an  $n$ -ary relation. From (18) and (21), it follows that there is a unique function  $\text{CUR}(f_R)$  such that:

$$< a_1, \dots, a_n > \in R \quad \text{iff} \quad \text{CUR}(f_R)(a_1) \dots (a_n) = 1$$

Note: Again, the lambda notation makes it quite easy to define  $\text{CUR}(f_R)$

$$\text{CUR}(f_R) = [ \lambda x_1 : \dots [ \lambda x_n : f_R(< x_1, \dots, x_n >) ] \dots ]$$

Note:

Putting all of this together, we will often shift freely between the following (distinct) set-theoretic objects:

- Set of  $n$ -tuples ( $n$ -ary relations)
- Functions from  $n$ -tuples to  $\{1, 0\}$  (characteristic functions of  $n$ -ary relations)
- Curried functions from  $n$ -tuples to  $\{1, 0\}$
- $A^{B \times C}$  and  $(A^C)^B$

## 5. Indexing

### (23) Indexed Family

- Throughout the notes above, I've made use of the following informal notation:

$$A = \{a_1, \dots, a_n\}$$

'A is a set consisting of n different elements  $a_i$ , for all  $0 < i \leq n$ '

- Note that this informal notation implies the existence of a bijection  $f: \{1, 2, \dots, n\} \rightarrow A$   
 $f(i) = a_i$
- Note that there's no special reason why we have to use numbers as indices; it's just convenient. This sets up the following general definition.

#### Definition:

Let J and A be sets such that there is a bijection  $f: J \rightarrow A$ . We can say that A is an *indexed family*, and that J is the *index set*.

#### Notation:

Suppose that A is an indexed family, whose index set is J. We can represent A as follows:

$$(i) \quad \{ a : \text{there is } a j \in J \text{ such that } f^{-1}(a) = j \}$$

$$(ii) \quad \{ a_j \}_{j \in J}$$

### (24) Indexing and Tuples

- Throughout the notes above, we've also used numerical indices to represent *n*-tuples:

$$< a_1, \dots, a_n >$$

- We can adapt our notation in (ii) above as means for compactly representing *n*-tuples.  
*Both of the following are equivalent to  $< a_1, \dots, a_n >$ :*

$$(i) \quad < a_i >_{i \in \{1, \dots, n\}}$$

$$(ii) \quad < a_i >_{i \leq n}$$

(25) **Union and Intersection of Indexed Sets**

Let S be an indexed family of sets with index set A.

a.  $\cup S = \{ x : x \in S_a \text{ for some } a \in A \}$

b.  $\cap S = \{ x : x \in S_a \text{ for all } a \in A \}$

(26) **Alternate Notations**

- a. If S is an indexed family of sets with index set A, then ' $\cup S$ ' is sometimes written:

$$\bigcup_{\alpha \in A} S_\alpha$$

- b. If S is an indexed family of sets with index set A, then ' $\cap S$ ' is sometimes written:

$$\bigcap_{\alpha \in A} S_\alpha.$$

- c. If S is an indexed family of sets with index set {1, ..., n}, then ' $\cup S$ ' is sometimes written:

$$\bigcup_{r=1}^n S_r$$

- c. If S is an indexed family of sets with index set {1, ..., n}, then ' $\cap S$ ' is sometimes written:

$$\bigcap_{r=1}^n S_r$$

## Formal Preliminaries, Part 2: Cardinalities, Infinities, and Proof by Induction<sup>1</sup>

### (1) Some Key Players

#### a. The Natural Numbers ( $\mathbb{N}$ )

- All the whole numbers greater than or equal to 0,  $\{0, 1, 2, \dots\}$

#### b. The Integers ( $\mathbb{Z}$ )

- All the whole numbers (including those less than 0),  $\{\dots, -2, -1, 0, 1, 2 \dots\}$

#### c. The Rational Numbers ( $\mathbb{Q}$ )

- All the numbers that can be written as a fraction,  $\{n/m : n, m \in \mathbb{Z} \text{ & } m \neq 0\}$

#### d. The Real Numbers ( $\mathbb{R}$ )

- All the numbers that can be represented by an infinite decimal expansion
- All the rational numbers and all the irrational numbers (e.g.,  $\pi$ )

Note:  $\mathbb{N} \subset \mathbb{Z} \subset \mathbb{Q} \subset \mathbb{R}$

### 1. Cardinalities and Infinities

(2) **Cardinality (Informal)**       $|A| =$       ‘the cardinality of A’  
   =      ‘the number of elements in A’

### (3) Cardinality, Injection, and Bijection

The following statements are intuitively true for finite sets. We'll therefore assume they are true for *all* sets (including infinite ones).

- a.  $|A| \leq |B|$       if and only if there is an **injection**  $f: A \rightarrow B$
- b.  $|A| = |B|$       if and only if there is an **bijection**  $f: A \rightarrow B$

Note:

- Recall that if  $f$  is a bijection, then  $f^{-1}$  is a bijection too.
- Thus, if there is a bijection  $f: A \rightarrow B$ , then there is also a bijection  $f^{-1}: B \rightarrow A$ 
  - Thus, by the definition in (3b), if  $|A| = |B|$ , then  $|B| = |A|$  (as desired)
  - Also, as you can prove to yourself:  
 If  $|A| = |B|$  and  $|B| = |C|$ , then  $|A| = |C|$   
 For any set  $A$ ,  $|A| = |A|$

---

<sup>1</sup> These notes are based upon the following required readings: Partee *et al.* (1993), Chapter 4, pp. 192-198.

(4) **Key Consequence: Infinite Sets Can Have Same Cardinality as Proper Subsets**

- Consider the following function:  $f(x) = 2x$  /  $f = \{ \langle x, y \rangle : y = 2x \}$
- This function  $f$  is a bijection from  $\mathbb{N}$  to the set of even numbers!
  - It's an injection: every  $x$  is mapped to a different even number
  - It's a surjection: every even number is equal to  $2x$  for some  $x \in \mathbb{N}$
- Thus, even though  $\{ n : n \in \mathbb{N} \text{ and } n \text{ is even} \} \subset \mathbb{N}$ ,  
$$|\{ n : n \in \mathbb{N} \text{ and } n \text{ is even} \}| = |\mathbb{N}|$$
- Intuitively, no finite set contains a proper subset of the same cardinality.
  - Thus, we can take this interesting property of  $\mathbb{N}$  as characteristic of ‘infinities’

(5) **Characterization of Non-Finite**

A set  $S$  is infinite *if and only if* there is a proper subset  $S' \subset S$  such that  $|S'| = |S|$

(6) **Transfinite Cardinals**

- Although it seems sensible to speak of  $|\mathbb{N}|$ , there is clearly no finite cardinal number  $n \in \mathbb{N}$  such that  $|\mathbb{N}| = n$ .
- It will be useful to introduce new, *transfinite* cardinal numbers to allow us give a name to the cardinality of  $\mathbb{N}$
- We introduce the special symbol ‘ $\aleph_0$ ’ (aleph null) below to refer to this first transfinite cardinal.

$$\aleph_0 = |\mathbb{N}|$$

(7) **Countable and Countably Infinite**

- a. A set  $S$  is **countably (denumerably) infinite** *iff*  $|S| = \aleph_0$
- b. A set  $S$  is **countable** *iff*  $S$  is finite or  $S$  is countably infinite.

(8) **Demonstrating that an Infinite Set is Countable, Part 1**

- To show that an infinite set  $S$  is countable, show that there is a bijection from  $S$  to  $\mathbb{N}$
- After all, this would entail  $|S| = |\mathbb{N}| = \aleph_0$

(9) **The Natural Numbers Without Zero ( $\mathbb{N} - \{0\}$ ) are Countable**

Consider the following function:  $f(n) = n - 1$

- a. The function  $f$  is clearly an injection from  $\mathbb{N} - \{0\}$  to  $\mathbb{N}$   
(each number in  $\mathbb{N} - \{0\}$  is mapped to a different member of  $\mathbb{N}$ )
- b. The function  $f$  is clearly a surjection from  $\mathbb{N} - \{0\}$  to  $\mathbb{N}$   
(every member of  $\mathbb{N}$  is equal to  $(n-1)$  for some element in  $\mathbb{N} - \{0\}$ )

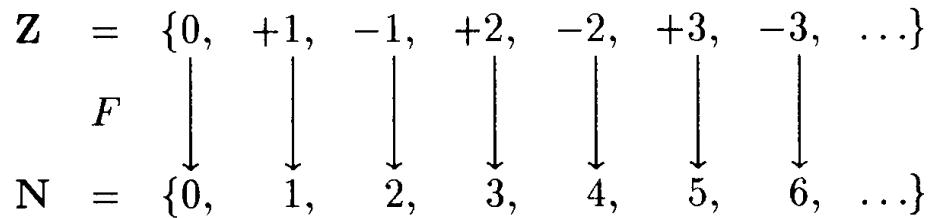
Thus,  $|\mathbb{N} - \{0\}| = |\mathbb{N}| = \aleph_0$

(10) **The Integers  $\mathbb{Z}$  are Countable**

Consider the function  $f$  defined below:

$$\begin{aligned} f(x) &= 0 && \text{if } x = 0 \\ && 2x-1 && \text{if } x \text{ is positive} \\ && -2x && \text{if } x \text{ is negative} \end{aligned}$$

*Picture of  $f(x)$ , from Partee et al. (1993):*



- a.  $f(x)$  is clearly a function from  $\mathbb{Z}$  to  $\mathbb{N}$
- b.  $f(x)$  is an injection from  $\mathbb{Z}$  to  $\mathbb{N}$ 
  - Each positive number is mapped to an odd number
  - Each negative number is mapped to an even number (greater than 0)
  - Only 0 is mapped to 0
- c.  $f(x)$  is a surjection from  $\mathbb{Z}$  to  $\mathbb{N}$ 
  - 0 is mapped to 0
  - Every positive even number is equal to  $-2x$  for some negative integer
  - Every positive odd number is equal to  $2x-1$  for some positive integer

Therefore,  $f(x)$  is a bijection, and so  $|\mathbb{Z}| = |\mathbb{N}| = \aleph_0$

(Note, this is despite the fact that  $\mathbb{N} \subset \mathbb{Z}$ )

### (11) Demonstrating that an Infinite Set is Countable, Part 2

Now that we know that  $\mathbb{Z}$  and  $\mathbb{N}-\{0\}$  are countable, we can show that  $S$  is countable by showing that there is a bijection from  $S$  to  $\mathbb{Z}$  or from  $S$  to  $\mathbb{N}-\{0\}$

- After all, this would entail  $|S| = |\mathbb{Z}| = \aleph_0$  or  $|S| = |\mathbb{N}-\{0\}| = \aleph_0$

### (12) The Positive Rationals are Countable

Usually, the following ‘intuitive’ (or ‘visual’) proof is used to show that there is a bijection from  $\{ n : n \in \mathbb{Q} \text{ and } n > 0 \}$  to  $\mathbb{N}-\{0\}$

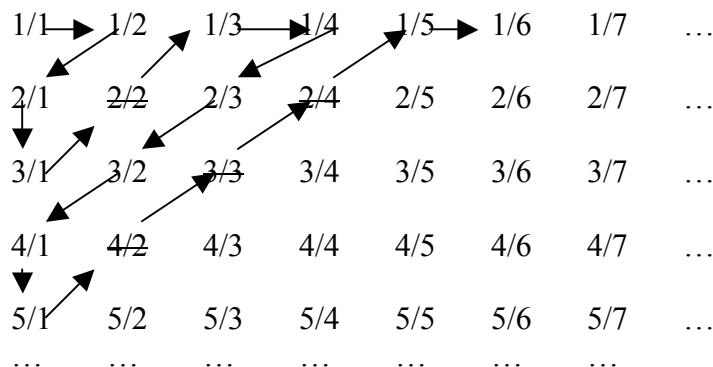
a. Step One:

We can arrange the set  $\{ n : n \in \mathbb{Q} \text{ and } n > 0 \}$  into the following infinite table:

1/1	1/2	1/3	1/4	1/5	1/6	1/7	...
2/1	2/2	2/3	2/4	2/5	2/6	2/7	...
3/1	3/2	3/3	3/4	3/5	3/6	3/7	...
4/1	4/2	4/3	4/4	4/5	4/6	4/7	...
...	...	...	...	...	...	...	...

b. Step Two:

Some rationals appear more than once in this table (e.g.,  $1/1 = 2/2$ ). We can fix this by snaking around the grid (infinitely) in the way sketched below. Every time we hit a number that we’ve already passed, we cross it out.

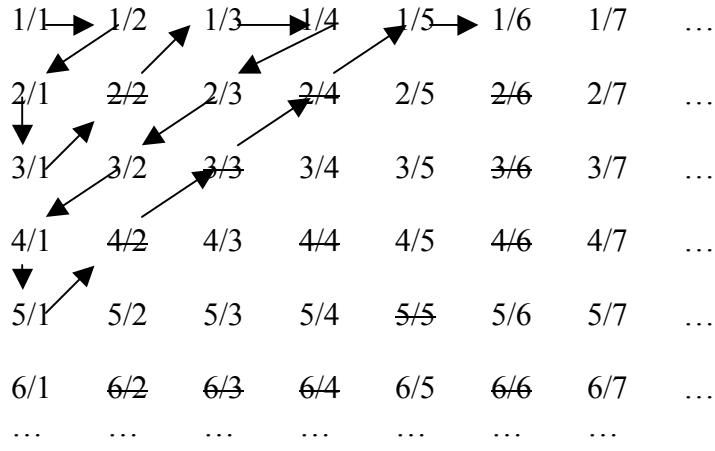


c. Step Three:

We take the augmented grid resulting from Step 2 (where repeated rationals are crossed out), and we snake through it again just as before, mapping the rationals in the grid to  $\mathbb{N}-\{0\}$  in the following way:

- (i) We map  $1/1$  to 1
- (ii) We then proceed as follows:

- Suppose that at step  $m$  in the ‘snaking’ we’ve just mapped the rational  $p/q$  to the natural number  $n$ .
- We next examine the rational  $r/s$  we come to at step  $(m+1)$ .
  - If  $r/s$  is *not* crossed off in the grid, we map it to  $(n+1)$ .
  - If  $r/s$  is crossed off, then we proceed to the next step in the snaking...



$f(1/1) =$	1	$f(2/3) =$	7
$f(1/2) =$	2	$f(3/2) =$	8
$f(2/1) =$	3	$f(4/1) =$	9
$f(3/1) =$	4	$f(5/1) =$	10
$f(1/3) =$	5	$f(1/5) =$	11
$f(1/4) =$	6	$f(1/6) =$	12
			...

d. Step Four:

The function  $f$  defined above is a bijection from  $\{ n : n \in \mathbb{Q} \text{ and } n > 0 \}$  to  $\mathbb{N}-\{0\}$

- (i) It is an injection:

No two rationals will end up mapped to the same number in  $\mathbb{N}-\{0\}$

- (ii) It is a surjection:

Since there are infinite number of positive rationals, every number in  $\mathbb{N}-\{0\}$  will be equal to  $f(x)$  for some rational  $x$ .

(13) **The Rationals are Countable**

- We can use the result in (12) to show that the entire set of rationals  $\mathbb{Q}$  is countable.
- Consider the function  $h$  defined below (where  $f$  is the function in (12))

$$\begin{aligned} \text{For all } n \in \mathbb{Q}, h(n) = & 0 && \text{if } n = 0 \\ & f(n) && \text{if } n > 0 \\ & -f(-n) && \text{if } n < 1 \end{aligned}$$

The function  $h$  above is a bijection from  $\mathbb{Q}$  to  $\mathbb{Z}$

- a. The function  $h$  is clearly an injection
  - Because  $f$  is an injection to  $\mathbb{N}-\{0\}$  every positive rational will be mapped to a different positive integer, and every negative rational will be mapped to a different negative integer.
- b. The function is clearly a surjection
  - Because  $f$  is a surjection to  $\mathbb{N}-\{0\}$  every positive integer  $n$  will be equal to  $h(x)$  ( $=f(x)$ ) for some positive rational.
  - Because  $f$  is a surjection to  $\mathbb{N}-\{0\}$  every negative integer  $n$  will be equal to  $h(x)$  ( $=-f(-x)$ ) for some negative rational).

Therefore,  $|\mathbb{Q}| = |\mathbb{Z}| = \aleph_0$

---

2. **Uncountable (Non-denumerable) Sets**

So far, we've seen that  $|\mathbb{N}| = |\mathbb{Z}| = |\mathbb{Q}| = \aleph_0$ , even though  $\mathbb{N} \subset \mathbb{Z} \subset \mathbb{Q}$

- This might lead one to wonder whether, in addition,  $|\mathbb{R}| = \aleph_0$
- In this section, we'll see that this is *not* the case,  $|\mathbb{R}| \neq \aleph_0$ 
  - That is, there are some infinities that are *uncountable (non-denumerable)*

(14) **Powersets and Cardinalities**

For every set  $S$ ,  $|S| < |\wp(S)|$

- Suppose that  $|S| = |\wp(S)|$ . There must then be a bijection  $f: S \rightarrow \wp(S)$
- Now, for any element  $x \in S$ , it is clear that either  $x \in f(x)$  or  $x \notin f(x)$
- Thus, we can define the set  $R = \{x : x \notin f(x)\}$
- Now,  $R$  is a subset of  $S$ , and so there must be some  $y \in S$  such that  $f(y) = R$
- Finally, it must be that either  $y \in f(y)$  or  $y \notin f(y)$ 
  - Suppose  $y \in f(y)$ . But then  $y \notin R$ , and so  $y \notin f(y)$ . Contradiction.
  - Suppose  $y \notin f(y)$ . But then  $y \in R$ , and so  $y \in f(y)$ . Contradiction.

(15) **Key Consequence:**  $|\wp(\mathbb{N})| > |\mathbb{N}| = \aleph_0$

If a set  $S$  is such that  $|S| > \aleph_0$ , then we say that  $S$  is *uncountable (non-denumerable)*.

(16) **Key Result: The Real Numbers Between 0 and 1 are Uncountable**

a. Key Background Fact:

Every real number between 0 and 1 can be uniquely represented as a sequence consisting of ‘0.’, followed by an infinitely long number of decimals:

0.13456789890989999....

0.57682838494827789...

- Thus, every real number between 0 and 1 uniquely corresponds to a sequence of the form ‘0. $a_1a_2a_3a_4a_5a_6\dots$ ’, where each  $a_i$  is a decimal numer.

b. The Proof:

- Suppose that  $|\{n : n \in \mathbb{R} \text{ and } 0 < n < 1\}| = |\mathbb{N}|$ . Then there is a bijection  $f$  from  $\{n : n \in \mathbb{R} \text{ and } 0 < n < 1\}$  to  $\mathbb{N}$ .
- Given this bijection  $f$ , it is possible to write an (infinitely long) list of all the members of  $\{n : n \in \mathbb{R} \text{ and } 0 < n < 1\}$ . Given the key background fact in (16a), this list will look as follows, where  $a_{nm}$  is the  $m^{\text{th}}$  decimal in the  $n^{\text{th}}$  real number in the ordering:

1	0. $a_{11}a_{12}a_{13}a_{14}a_{15}\dots$
2	0. $a_{21}a_{22}a_{23}a_{24}a_{25}\dots$
3	0. $a_{31}a_{32}a_{33}a_{34}a_{35}\dots$
4	0. $a_{41}a_{42}a_{43}a_{44}a_{45}\dots$
...	

- Now, we can use this list to define a real number  $r$  between 0 and 1 that is *not* on this list:
  - The integer component of  $r$  is 0
  - The first decimal in  $r$  after 0 is different from  $a_{11}$
  - The second decimal in  $r$  after 0 is different from  $a_{22}$
  - The third decimal in  $r$  after 0 is different from  $a_{33}$
  - (and so on...)
- The real number  $r$  is guaranteed not to appear anywhere on this list.
  - After all, for any natural number  $n$ ,  $r$  will differ from  $f(n)$  in the  $n^{\text{th}}$  decimal after 0.
- Therefore, this list *doesn't* contain all the real numbers between 0 and 1. Consequently, there is no bijection from  $\{n : n \in \mathbb{R} \text{ and } 0 < n < 1\}$  to  $\mathbb{N}$ .

**Thus,  $|\{n : n \in \mathbb{R} \text{ and } 0 < n < 1\}| \neq |\mathbb{N}|$ . Thus,  $|\mathbb{R}| > |\mathbb{N}| = \aleph_0$**

### (17) Additional Transfinite Cardinals

- For various reasons, it will be helpful to have a name for  $|\wp(\mathbb{N})|$ :

$$2^{\aleph_0}$$

'the cardinality of  $\wp(\mathbb{N})$ '

- It is known that  $|\mathbb{R}| = |\wp(\mathbb{N})|$
- 
- 

### 3. Proof by Mathematical Induction

#### (18) Key Axiom of Number Theory

Suppose that for some property  $P$ , we can show (i) and (ii):

- (i) 0 has property  $P$
- (ii) For any  $n \in \mathbb{N}$ , if  $n$  has property  $P$ , then  $(n+1)$  has property  $P$ .

Then we can conclude that *every*  $n \in \mathbb{N}$  has property  $P$

#### (19) Key Consequence of (18)

Suppose that for some property  $P$ , we can show (i) and (ii):

- (i) 0 has property  $P$
- (ii) For any  $n \in \mathbb{N}$ , if *every* number  $m < n$  has property  $P$ , then  $n$  has  $P$

Then we can conclude that *every*  $n \in \mathbb{N}$  has property  $P$

#### (20) Some Terminology

- a. An argument making use of the axiom in (18) is typically referred to as a *proof by (weak) induction*
- b. An argument making use of the consequence in (19) is typically referred to as *proof by strong induction*.
- c. In a proof by (weak/strong) induction,
  - (i) Proving that 0 has property  $P$  is called the 'base step' ('base case')
  - (ii) Proving either (18ii) or (19ii) is called the 'induction step'.

**Note:** If the base case is some numeral  $n > 0$ , then a proof by induction demonstrates that  $P$  holds for all  $m$  such that  $n \leq m$

(21) **Key Application**

If  $S$  is a countable set – that is, if there is a bijection  $f: \mathbb{N} \rightarrow S$  – then we can use proofs by induction to prove things about  $S$ !

(22) **Illustration: Generalized Distributive Law**

Claim:

For all  $n \in \mathbb{N}$  such that  $2 \leq n$ ,  $A \cup (B_1 \cap \dots \cap B_n) = (A \cup B_1) \cap \dots \cap (A \cup B_n)$

Proof by Induction:

a. *Base Step:  $n = 2$*

$$A \cup (B_1 \cap B_2) = (A \cup B_1) \cap (A \cup B_2)$$

This follows from the simple set-theoretic equivalences proven in Chapter 1 of Partee *et al.* (1993).

b. *Induction Step*

Let  $n \in \mathbb{N}$  be such that:  $A \cup (B_1 \cap \dots \cap B_n) = (A \cup B_1) \cap \dots \cap (A \cup B_n)$

o By the associativity of intersection:

$$A \cup (B_1 \cap \dots \cap B_n \cap B_{n+1}) = A \cup ((B_1 \cap \dots \cap B_n) \cap B_{n+1})$$

o Next, by the base step in (22a):

$$A \cup ((B_1 \cap \dots \cap B_n) \cap B_{n+1}) = (A \cup (B_1 \cap \dots \cap B_n)) \cap (A \cup B_{n+1})$$

o Next, by the induction assumption for  $n$ :

$$(A \cup (B_1 \cap \dots \cap B_n)) \cap (A \cup B_{n+1}) = ((A \cup B_1) \cap \dots \cap (A \cup B_n)) \cap (A \cup B_{n+1})$$

o Finally, by the associativity of intersection again:

$$((A \cup B_1) \cap \dots \cap (A \cup B_n)) \cap (A \cup B_{n+1}) = (A \cup B_1) \cap \dots \cap (A \cup B_n) \cap (A \cup B_{n+1})$$

o **Thus,**  $A \cup (B_1 \cap \dots \cap B_n \cap B_{n+1}) =$

$$(A \cup B_1) \cap \dots \cap (A \cup B_n) \cap (A \cup B_{n+1})$$

**Therefore, by (weak) induction, it follows that for all  $n \in \mathbb{N}$  such that  $2 \leq n$ :**

$$A \cup (B_1 \cap \dots \cap B_n) = (A \cup B_1) \cap \dots \cap (A \cup B_n)$$

(23) **Illustration of Strong Induction: Well Ordering Principle**

Claim: If  $S \subseteq \mathbb{N}$  and  $S \neq \emptyset$ , then there is an  $a \in S$  such that for all  $s \in S$ ,  $a \leq s$ .

Proof by Strong Induction:

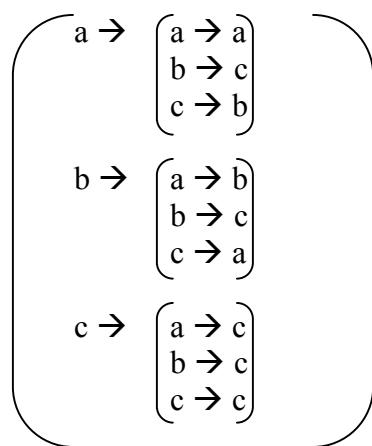
- Suppose that there is an  $S \subseteq \mathbb{N}$  and  $S \neq \emptyset$ . For a contradiction, suppose that there is *no*  $a \in S$  such that for all  $s \in S$ ,  $a \leq s$ .
- By strong induction, we'll show that for all  $n \in \mathbb{N}$ ,  $n \notin S$ , and so  $S = \emptyset$ , contrary to assumption.
  - a. *Base Step: n = 0*  
Clearly,  $0 \notin S$ . (After all, for all  $s \in S$ ,  $0 \leq s$ )
  - b. *Induction Step*  
Let  $n \in \mathbb{N}$  be such that for all  $m < n$ ,  $m \notin S$ . We will show that  $n \notin S$ .
    - Suppose that  $s \in S$ . Now, clearly  $(n-1) < s$ . (After all, if  $s \leq (n-1)$ , then  $s < n$ , and so by the induction assumption  $s \notin S$ , contrary to assumption.)
    - Next, since  $(n-1) < s$ , it follows that  $n \leq s$ . Since  $s$  was arbitrary, it follows that for all  $s \in S$ ,  $n \leq s$ .
    - **Consequently,  $n \notin S$**  (After all, by assumption there is no  $a \in S$  such that for all  $s \in S$ ,  $a \leq s$ .)
  - **Thus, by strong induction, for all  $n \in \mathbb{N}$ ,  $n \notin S$ , and so  $S = \emptyset$ , contrary to assumption.**
  - **Therefore, for any  $S \subseteq \mathbb{N}$  and  $S \neq \emptyset$ , there is  $a \in S$  such that for all  $s \in S$ ,  $a \leq s$ .**

### Problem Set on the ‘Formal Preliminaries’<sup>1</sup>

#### (1) Basic Comprehension Questions on Relations and Functions

- a. Let  $A = \{b,c\}$  and  $B = \{2,3\}$ . State whether the following are true or false:
  - (i)  $(A \times B) \cap (B \times A) = \emptyset$
  - (ii)  $\langle c, c \rangle \subseteq A^2$
  - (iii)  $\{\langle b, 3 \rangle, \langle 2, a \rangle\} \subseteq (A \times B) \cup (B \times A)$
  - (iv)  $\emptyset \subseteq (A \times A)$
- b. Let  $A = \{b,c\}$  and  $B = \{2,3\}$ . Let  $R = \{\langle b,b \rangle, \langle b,2 \rangle, \langle c,2 \rangle, \langle c,3 \rangle\}$ 
  - (i) What is the range and domain of  $R$ ?
  - (ii) What is  $R^{-1}$ ?
- c. Let  $f: A \rightarrow B$  and  $g: B \rightarrow C$  both be *bijections*. Show that  $(g \circ f)^{-1} = (f^{-1}) \circ (g^{-1})$
- d. Let  $R = \{ \langle x,y \rangle : x, y \in \{1, 2\} \text{ and } x < y \}$ .
  - (i) Represent  $R$  as a set of pairs.
  - (ii) Represent the characteristic function of  $R$  as a set of pairs<sup>2</sup>
  - (iii) Represent the characteristic function of  $R$  as a ternary relation
  - (iv) Represent the curried characteristic function of  $R$  as a matrix

*Reminder: By ‘matrix’ I mean a diagram like the following:*




---

<sup>1</sup> Most of these exercises are taken from Partee *et al.* (1993), Chapters 2 and 4.

<sup>2</sup> Assume that the domain of the characteristic function of  $R$  is  $\{1,2\} \times \{1,2\}$ .

## (2) Proving that Sets are Countable

- a. Show that the set of integer powers of ten {10, 100, 1000, 10000, ... } is countable.
- b. Suppose that the following is true of English:
  - (i) There is a finite alphabet for writing sentences, consisting of 26 letters and a space (forget punctuation marks for now)
  - (ii) Every sentence of English is a finite string in the alphabet in (i)
  - (iii) There is no upper bound on the length of sentences of English. That is, for any sentence S of English, there is always a longer sentence S'.

Show that the set of English sentences is countably infinite.

### Some Hints:

- To solve this, you simply have to show that there is a defined way of ordering the sentences of English into an infinite ‘list’ (as we did for the rationals greater than 0).
- One way of solving this makes use of ‘alphabetical order’.
- Another, more round-about way of solving this makes use of the following key consequence of the ‘fundamental theorem of arithmetic’

### Consequence of Fundamental Theorem of Arithmetic:

For every positive integer  $n > 1$ , there is *exactly one* way of representing  $n$  as a product of powers of primes:

$$n = p_1^{a_1} \times p_2^{a_2} \times \dots \times p_k^{a_k} \quad \text{where each } p_i \text{ is prime, each } a_i \in \mathbb{N}$$

## (3) Proofs by Induction

Construct a proof by induction for the following general equivalence:

$$(X_1 \cup \dots \cup X_n)' = X_1' \cap \dots \cap X_n'$$

### Hint:

You can appeal to the following general equivalences:

- (i)  $X'' = X$
- (ii)  $X_1 \cap \dots \cap X_n = (X_1 \cap \dots \cap X_{(n-1)}) \cap X_n$
- (iii)  $X_1 \cup \dots \cup X_n = (X_1 \cup \dots \cup X_{(n-1)}) \cup X_n$

**Unit 2:**  
**Review of Propositional Logic and First Order Logic**

## Propositional Logic: Syntax and Natural Deduction <sup>1</sup>

### The Plot That Will Unfold

- I want to provide some key historical and intellectual context to the ‘model theoretic’ approach to natural language semantics, context that is usually missing from introductory semantics classes / texts.
- I want to begin by getting across the very motivation for having ‘models’ at all:
  - Why did logicians develop this mathematical device?
  - What is it good for? What did it originally do for us?
- Consequently, I’m going to start off by giving a *purely syntactic* introduction to the systems of Propositional Logic and First Order (Predicate) Logic.
  - We’ll then see that certain key questions about these systems will require us to develop a mathematically precise characterization of what it is to be an ‘interpretation’ of these formal languages...

### 1. Some Historical Background

#### (1) Valid Argument

An argument is *valid* if whenever the premises are true, the conclusion *must* be true.

- a. **Valid Argument:** All men are mortal. Socrates is a man.  
Therefore, Socrates is mortal.
- b. **Invalid Argument:** All cats are mortal. All dogs are mortal.  
Therefore, all cats are dogs.

#### (2) Key Insight from Ancient Philosophers (Aristotle)

Many valid arguments seem to share the same general *syntactic form*.

- a. **Valid Arguments:** All Ps are Qs. S is a P.  
Therefore, S is a Q.  
(Barbara) No Ps are Qs. All Ss are Ps.  
Therefore, No Ss are Qs.  
(Celarent)
- b. **Aristotelian Logic (Syllogistic Logic):**  
First to characterize valid arguments purely in terms of their *syntactic form*.
  - So successful that it was logic from 300BC to 1800s.

---

<sup>1</sup> These notes are based upon material in the following required reading: Gamut (1991), Chapter 1, Chapter 2 pp. 28-41, Chapter 4 pp. 128-141; Partee et al. (1993) Chapter 5, Chapter 6 pp. 97-98.

(3) **Major Weakness of Syllogistic Logic**

- Only applies to ‘syllogisms’, a very restricted form of argument.
- Consequently, cannot capture valid arguments based on *relations*:

All horses are animals.

Therefore, every horse’s head is an animal’s head.

(4) **Key Developments in the 19<sup>th</sup> Century**

a. Early-to-Mid 1800’s

- Renewed activity in logic; first highly original work in centuries (George Boole, Augustus DeMorgan)
- Renewed interest in whether mathematics derives from ‘pure logic’.

b. Frege’s *Begriffsschrift* (1879)

- Birth of modern formal logic.
- First appearance of quantified variables
- Essentially all of First Order Logic (with a little Second Order Logic, too)
- Highly idiosyncratic 2-dimensional notation<sup>2</sup>
- Reconstructs an unprecedented amount of mathematics and natural language reasoning
  - First real indication that math could just derive from pure logic

c. Further Developments After *Begriffsschrift*

- Other logicians develop and modify the key insights of *Begriffsschrift*
- Peano introduces many of our modern logical notations:  $\forall$ ,  $\exists$ ,  $\supset$
- **Principia Mathematica (1910-1913; but begun much earlier)**
  - Could be called the ‘climax’ of the 19<sup>th</sup> century period
  - Three-volume work reconstructing enormous amounts of mathematics from a logical proof system close to what we have today.

---

<sup>2</sup> For a nice overview of Frege’s funky logical notation: <http://en.wikipedia.org/wiki/Begriffsschrift>

## (5) The Core Idea Behind a Formal Logic

- A precisely defined formal notation for representing *certain aspects* (not all) of the ‘logical structure’ of an assertion.
- A set of *syntactically defined* rules for deriving formulas in the notation from other formulas in the notation.

Illustration: Natural Deduction (Logic 101):

1.	$p \rightarrow (\neg q \vee r)$	Premise
2.	$q$	Premise
3.	$q \rightarrow \neg r$	Premise
4.	$\neg r$	Modus Ponens 2,3
5.	$q \& \neg r$	Conjunction 2,4
6.	$\neg(\neg q \& \neg r)$	Double Negation 5
7.	$\neg(\neg q \vee \neg r)$	DeMorgans 6
8.	$\neg(\neg q \vee r)$	Double Negation 7
9.	$\neg p$	Contraposition 1,8

## (6) An Immediate Problem for This Program (Foreshadowing)

- As with classical logic, the goal of the enterprise is to provide a purely *syntactic* characterization of what it is to be a *valid argument*.
- But, the notion of ‘validity’ is a *semantic* one.  
(if premises *true*, then conclusion must be *true*)
- **So, how can you be sure that your syntactically defined system does what it’s supposed to do?**
  - How do you show that it *only* derives valid inferences?
  - How do you show that it derives *all* the valid inferences?
- **To do this, you need a mathematically precise characterization of what it means for a formula to be ‘true’ (relative to an ‘interpretation’).**
  - It wasn’t until about the 1930’s or so until this idea reached its modern form (Löwenheim, Gödel, Tarski)

*Let’s begin by putting ourselves in the position of a person in 1910 or so...*

*We’ll start off with a purely syntactic introduction to Propositional Logic*

## 2. A Review of Propositional Logic (PL): Syntax and Informal Semantics<sup>3</sup>

The system of Propositional Logic (PL) is intended to capture the inferences that depend upon the meaning of the so-called ‘sentential connectives’: *and*, *or*, *if...then*, and *not*

### (7) The Vocabulary of Symbols

- a. The Non-Logical Constants (a.k.a ‘The Logical Variables’)  
An infinite set of *proposition letters*: {p, q, r, s, t, … p<sub>1</sub>, p<sub>2</sub>, p<sub>3</sub>, p<sub>4</sub>, …}
- b. The Logical Constants:

(i)	~	Negation	‘It is not the case that…’
(ii)	&	Conjunction	‘and’
(iii)	v	Disjunction	‘or’ (inclusive)
(iv)	→	(Material) Implication	‘if…then’
- c. Syntactic Symbols: ( , )

### (8) The Definition of a ‘Well-Formed Formula’ (WFF) of PL

The set of ‘well-formed formulae’ of PL, WFF, is the smallest set such that:

- a. If  $\varphi$  is a proposition letter, then  $\varphi \in \text{WFF}$
- b. If  $\varphi, \psi \in \text{WFF}$ , then
  1.  $\sim\varphi \in \text{WFF}$
  2.  $(\varphi \ \& \ \psi) \in \text{WFF}$
  3.  $(\varphi \ v \ \psi) \in \text{WFF}$
  4.  $(\varphi \rightarrow \psi) \in \text{WFF}$

### (9) Using PL To Encode Sentences of English

- We can use the syntactic rules in (8) and the informal semantics in (7) to write PL formulae that ‘encode’ certain statements of English:
  - a. *Sentence:* “Dave is tall, and if Dave isn’t tall, then Mary is dancing.”  
*Encoding:* ( p & ( ~p → q ) )
  - b. *Sentence:* “If Bill or John is leaving, then Mary and Sue are not happy.”  
*Encoding:* ( ( b v j ) → ( ~m & ~s ) )
- In setting up such encodings, it is critical to supply a ‘key’, indicating which English assertions the propositional letters ‘stand for’: 

c. <i>Key:</i>	p: Dave is tall	b: Bill is leaving	m: Mary is happy
	q: Mary is dancing	j: John is leaving	s: Sue is happy

<sup>3</sup> My discussion here will assume prior familiarity with the overall system of Propositional Logic. Students are referred to Partee *et al.* (1993), Chapter 6 for crucial background.

(10) **An Important Property of the PL Syntax**

- In (8), we provide a *recursive definition* of the well-formed formulae of PL.
- This definition allows us to use *mathematical induction* to prove things about *all* well-formed formulae of PL.

(11) **Illustration: An Inductive Proof about PL Formulae**

- a. Claim: Every well-formed formulae of PL has an even number of parentheses.  
(*i.e.*, If  $\varphi \in \text{WFF}$ , then there are an even number of parentheses in  $\varphi$ )
- b. Proof By Strong Induction  
We'll prove the claim by mathematical induction *on the number of logical constants in  $\varphi$* 
  - (i) **Base Step: 0**  
If  $\varphi$  contains no logical constants, then  $\varphi$  is a proposition letter, and so  $\varphi$  contains 0 parentheses (and 0 is even).
  - (ii) **Induction Step**  
Let  $n$  be such that for all  $m < n$ , if  $\varphi$  contains  $m$  logical constants, then  $\varphi$  has an even number of parentheses.
    - Now suppose that  $\varphi$  contains  $n$  logical constants. There are four possible cases to consider:
      1.  $\varphi$  is of the form  $\sim\psi$ , where  $\psi$  contains  $(n-1)$  logical constants. By assumption, then,  $\psi$  contains an even number of parentheses. Thus, so does  $\varphi$ .
      2.  $\varphi$  is of the form  $(\chi \ \& \ \psi)$ , where  $\chi$  contains  $m < n$  logical constants, and  $\psi$  contains  $j < n$  logical constants. By assumption, then,  $\chi$  and  $\psi$  both contain an even number of parentheses. Thus, so does  $\varphi$ .
      3.  $\varphi$  is of the form  $(\chi \vee \psi)$ , where  $\chi$  contains  $m < n$  logical constants, and  $\psi$  contains  $j < n$  logical constants. By argument parallel to (2), so does  $\varphi$ .
      4.  $\varphi$  is of the form  $(\chi \rightarrow \psi)$ , where  $\chi$  contains  $m < n$  logical constants, and  $\psi$  contains  $j < n$  logical constants. By argument parallel to (2), so does  $\varphi$ .

### (12) Proof by Induction on the Complexity of Formula

- The general structure of the proof in (11) is *extremely* common in (meta)logic. It's often called *proof by induction on the complexity of formula*.
- As the example in (11) makes clear, such 'proofs by induction on complexity of formula' are really just a special case of 'proof by (mathematical) induction'

#### Induction on the Complexity of Formula:

Suppose that you've shown both (i) and (ii) below. It follows that every formula  $\varphi$  in the language has property P.

- (i) Every primitive formula of the language has property P.
- (ii) For every complex formula  $\varphi$ , if the immediate subformulae of  $\varphi$  have property P, then so does  $\varphi$

## 3. A Review of Propositional Logic (PL): Natural Deduction

### (13) Major Goal of This Section

Let's provide a *purely syntactic* characterization of 'valid inference' in the PL notation.

- This syntactic characterization will be embodied in a *proof system* (natural deduction)
- We're going to lay out some rules – stated entirely in *syntactic terms* – for deriving formulae in PL from other formulae.
  - As we'll see, these syntactic rules intuitively capture certain key aspects of the everyday meaning of the English logical words 'not', 'and', 'or', 'if..then'

### (14) Definition of a Derivation

A *derivation* is a finite, numbered list of formulae, where each formula is accompanied by a coded statement indicating how it entered the derivation (see (5))

Such 'coded statements' will be of the following two forms:

- (i) 'Assumption' indicates that the formula is an assumption (premise)
- (ii) 'RULE n, ..., m' Where 'RULE' is the name of one of the derivation rules below, and 'n, ..., m' are the numbers of the formulae in the derivation that are 'input' to the rule

(15) Some Terminology and Notation

- a. Assumptions of the Derivation:  
The formulae in the derivation that are accompanied by ‘Assumption’
- b. Conclusion of the Derivation: The final line in the derivation.
- c. Key notation:  $\{ \varphi_1, \varphi_2, \varphi_3 \dots \} \vdash \psi$   
‘There is a derivation where *a subset of*  $\{ \varphi_1, \varphi_2, \varphi_3 \dots \}$  are the assumptions, and  $\psi$  is the conclusion’

**Note:** Given the way ‘ $\vdash$ ’ is defined in (15c), the set  $\{ \varphi_1, \varphi_2, \varphi_3 \dots \}$  could be *infinite*. This will make certain proofs about the system easier later on...

Now let's get to stating and illustrating some of the deduction rules!...

(16) The Rule of ‘&-Introduction’ (I&)

The following is an acceptable derivation:

1.	...	
...	...	
n <sub>1</sub>	φ	
...	...	
n <sub>2</sub>	ψ	
...	...	
m	(φ & ψ)	I& n <sub>1</sub> , n <sub>2</sub>

Illustration:  $\{ p, q, r \} \vdash ((p \& q) \& (p \& r))$

1.	p	Assumption
2.	q	Assumption
3.	r	Assumption
4.	(p & q)	I& 1,2
5.	(p & r)	I& 1,3
6.	((p & q) & (p & r))	I& 4,5

Intuitive Motivation:

If we can (in English) assert some sentence ‘S<sub>1</sub>’, and some sentence ‘S<sub>2</sub>’, then we can also assert the sentence ‘S<sub>1</sub> and S<sub>2</sub>’

Note:

From now on, I'll leave ‘the following is an acceptable derivation’ implicit in the statement of the deduction rules...

(17) **The Rule of '&-Elimination' (E&)**

1.	...		1.	...	
...	...		...	...	
n	$(\varphi \ \& \ \psi)$		n	$(\varphi \ \& \ \psi)$	
...	...		...	...	
m	$\varphi$	E& n	m	$\psi$	E& n

Intuitive Motivation:

If we can (in English) assert some sentence ' $S_1$  and  $S_2$ ', then we can also assert ' $S_1$ '.  
If we can (in English) assert some sentence ' $S_1$  and  $S_2$ ', then we can also assert ' $S_2$ '.

Illustration: Proving the Associativity of '&' (Part 1)

1.	$(p \ \& \ (q \ \& \ r))$	Assumption
2.	$p$	E& 1
3.	$(q \ \& \ r)$	E& 1
4.	$q$	E& 3
5.	$(p \ \& \ q)$	I& 2,4
6.	$r$	E& 3
7.	$((p \ \& \ q) \ \& \ r)$	I& 5,6

Note: By a similar derivation, we can show  $((p \ \& \ q) \ \& \ r) \vdash (p \ \& \ (q \ \& \ r))$

(18) **The Rule of 'Repetition'**

The following rule doesn't add anything interesting, but it makes proofs easier.

1.	...	
...	...	
n.	$\varphi$	
...	...	
m.	$\varphi$	Repetition n

Illustration:

1.	$(p \ \& \ (q \ \& \ r))$	Assumption
2.	$p$	E& 1
3.	$(q \ \& \ r)$	E& 1
4.	$q$	E& 3
5.	$r$	E& 3
6.	$(p \ \& \ q)$	I& 2,4
7.	$r$	Repetition 5
8.	$((p \ \& \ q) \ \& \ r)$	I& 6,7

(19) **The Rule of ' $\rightarrow$ -Elimination' ( $E\rightarrow$ )**

The following derivation rule is often referred to as ‘modus ponens’

1.	...	
...	...	
n <sub>1</sub>	( $\varphi \rightarrow \psi$ )	
...	...	
n <sub>2</sub>	$\varphi$	
...	...	
m	$\psi$	$E\rightarrow n_1, n_2$

Intuitive Motivation:

If we can (in English) assert the sentence ‘If S<sub>1</sub> then S<sub>2</sub>’, and we can assert the sentence ‘S<sub>1</sub>’, then we can also assert ‘S<sub>2</sub>’

Illustration: { (p  $\rightarrow$  (q  $\rightarrow$  r)), p, q }  $\vdash r$

1.	(p $\rightarrow$ (q $\rightarrow$ r))	Assumption
2.	p	Assumption
3.	q	Assumption
4.	(q $\rightarrow$ r)	$E\rightarrow 1,2$
5.	q	Repetition 3
6.	r	$E\rightarrow 4,5$

(20) **The Rule of ' $\rightarrow$ -Introduction' ( $I\rightarrow$ ) and ‘Conditional Proof’**

Intuitive Motivation:

If whenever we assume ‘S<sub>1</sub>’ it follows that ‘S<sub>2</sub>’, then we can assert ‘if S<sub>1</sub> then S<sub>2</sub>’.

The Rule:

The following is an acceptable derivation, as long as no line j > m makes reference to any lines from n<sub>1</sub> to n<sub>2</sub>.

1.	...	
...	...	
n <sub>1</sub>		$\varphi$ Assumption
...		
n <sub>2</sub>		$\psi$
m		( $\varphi \rightarrow \psi$ ) $I\rightarrow$

- In such a derivation, we say that the formulae occurring on lines n<sub>1</sub> to n<sub>2</sub> have been *dropped* or *withdrawn*
- Note the key restriction that once a formula has been *dropped*, no subsequent lines can make reference to it.

Illustration:  $((p \& q) \rightarrow r) \vdash (p \rightarrow (q \rightarrow r))$

1.	$((p \& q) \rightarrow r)$	Assumption
2.	p	Assumption
3.		Assumption
4.		I& 2,3
5.		E $\rightarrow$ 1, 4
6.	(q $\rightarrow$ r)	I $\rightarrow$
7.	$(p \rightarrow (q \rightarrow r))$	I $\rightarrow$

(21) **The Rule of ‘v-Introduction’ (Iv)**

1.	...	1.	...
...	...	...	...
n.	$\varphi$	n.	$\psi$
...	...	...	...
m.	$(\varphi \vee \psi)$	Iv n	$(\varphi \vee \psi)$
			Iv n

Intuitive Motivation:

If we can (in English) assert a sentence ‘S<sub>1</sub>’, then for any sentence ‘S<sub>2</sub>’, we can assert ‘S<sub>1</sub> or S<sub>2</sub>’ and ‘S<sub>2</sub> or S<sub>1</sub>’ (inclusive ‘or’)

(22) **The Rule of ‘v-Elimination’ (Ev)**

1.	...	1.	...
...	...	...	...
n <sub>1</sub>	$(\varphi \vee \psi)$	n.	$\psi$
...	...	...	...
n <sub>2</sub>	$(\varphi \rightarrow \chi)$		
...	...		
n <sub>3</sub>	$(\psi \rightarrow \chi)$		
...	...		
m	$\chi$	Ev n <sub>1</sub> , n <sub>2</sub> , n <sub>3</sub>	

Intuitive Motivation:

If we can (in English) assert ‘S<sub>1</sub> or S<sub>2</sub>’, ‘if S<sub>1</sub> then S<sub>3</sub>’ and also ‘if S<sub>2</sub> then S<sub>3</sub>’, then we can also assert ‘S<sub>3</sub>’

(23) **Illustration of the Rules for Disjunction: Associativity of ‘v’ (Part 1)**

$$(p \vee (q \vee r)) \vdash ((p \vee q) \vee r)$$

1.	$(p \vee (q \vee r))$	Assumption
2.	 p	Assumption
3.	(p $\vee$ q)	Iv 2
4.	((p $\vee$ q) $\vee$ r)	Iv 3
5.	$(p \rightarrow ((p \vee q) \vee r))$	$I\rightarrow$
6.	(q $\vee$ r)	Assumption
7.	q	Assumption
8.	(p $\vee$ q)	Iv 7
9.	((p $\vee$ q) $\vee$ r)	Iv 8
10.	$(q \rightarrow ((p \vee q) \vee r))$	$I\rightarrow$
11.	r	Assumption
12.	((p $\vee$ q) $\vee$ r)	Iv 11
13.	$(r \rightarrow ((p \vee q) \vee r))$	$I\rightarrow$
14.	((p $\vee$ q) $\vee$ r)	Ev 6, 10, 13
15.	$((q \vee r) \rightarrow ((p \vee q) \vee r))$	$I\rightarrow$
16.	$((p \vee q) \vee r)$	Ev 1, 5, 15

(24) **The Rule of ‘~ -Elimination’ (E~)**

For various reasons, it will help to have a special symbol indicating that a contradiction has been reached:  $\perp$  ‘falsum’

1.	...	
...	...	
n <sub>1</sub>	φ	
...	...	
n <sub>2</sub>	~φ	
...	...	
m	$\perp$	E~ n <sub>1</sub> n <sub>2</sub>

(25) **The Rule of ‘~ -Introduction’ (I~)**

With this special symbol, we can now state the following rule of I~

1.	...	
...	...	
n <sub>1</sub>	φ	Assumption
...	$\perp$	
n <sub>2</sub>	~φ	
m	$\perp$	I~

Intuitive Motivation:

If whenever we assume ‘S’ a contradiction follows, then we can assert ‘not S’.

(26) **Illustration of the Rules for Negation: Double Negation (Part 1)**

$$p \vdash \sim\sim p$$

1.	p	Assumption
2.		$\sim p$ Assumption
3.		$\perp$ E~ 1,2
4.	$\sim p$	I~

(27) **The Rule of ‘Ex Falso Sequitur Quodlibet’ (EFSQ)**

Intuitive Motivation: Anything follows from a contradiction.

1.	...	
...	...	
n	$\perp$	
m	$\varphi$	EFSQ

Illustration:  $\{ (p \vee q), \sim p \} \vdash q$

1.	$(p \vee q)$	Assumption
2.	$\sim p$	Assumption
3.		p Assumption
4.		$\perp$ E~ 2,3
5.		q EFSQ
6.	$(p \rightarrow q)$	I $\rightarrow$
7.		q Assumption
8.		q Repetition 7
9.	$(q \rightarrow q)$	I $\rightarrow$
10.	q	E $\vee$ 1, 6, 9

(28) **The Rule of ‘Double Negation’ ( $\sim$ )**

1.	...	
...	...	
n	$\sim\sim \varphi$	
...	...	
m	$\varphi$	$\sim\sim n$

Note:

Although our system without (28) can prove one half of the equivalence ‘ $\varphi$  iff  $\sim\sim\varphi$ ’ (26), we need the rule in (28) for our system to derive the complete equivalence.

#### 4. The Power of Our Natural Deduction System for PL

Although our system has just 11 (really, 10) relatively simple rules, it can capture a great many intuitively valid inferences!

##### (29) Derivation of DeMorgans, Part 1 $\sim(p \ \& \ q) \vdash (\sim p \vee \sim q)$

1.	$\sim(p \ \& \ q)$	Assumption
2.	$\sim(\sim p \vee \sim q)$	Assumption
3.	p	Assumption
4.	q	Assumption
5.	$(p \ \& \ q)$	I& 3,4
6.	$\perp$	E $\sim$ 1, 5
7.	$\sim q$	I $\sim$
8.	$(\sim p \vee \sim q)$	I $\vee$ 7
9.	$\perp$	E $\sim$ 2, 8
10.	$\sim p$	I $\sim$
11.	$(\sim p \vee \sim q)$	I $\vee$ 10
12.	$\perp$	E $\sim$ 2, 11
13.	$\sim\sim(\sim p \vee \sim q)$	I $\sim$
14.	$(\sim p \vee \sim q)$	$\sim\sim$ 13

##### (30) Derivation of DeMorgans, Part 2 $(\sim p \vee \sim q) \vdash \sim(p \ \& \ q)$

1.	$(\sim p \vee \sim q)$	Assumption
2.	$\sim p$	Assumption
3.	$(p \ \& \ q)$	Assumption
4.	p	E& 3
5.	$\perp$	E $\sim$ 2, 4
6.	$\sim(p \ \& \ q)$	I $\sim$
7.	$(\sim p \rightarrow \sim(p \ \& \ q))$	I $\rightarrow$
8.	$\sim q$	Assumption
9.	$(p \ \& \ q)$	Assumption
10.	q	E& 9
11.	$\perp$	E $\sim$ 8,10
12.	$\sim(p \ \& \ q)$	I $\sim$
13.	$(\sim q \rightarrow \sim(p \ \& \ q))$	I $\rightarrow$
14.	$\sim(p \ \& \ q)$	E $\vee$ 1, 7, 13

##### (31) The Big Question

Does our system offer a *perfect* syntactic characterization of ‘validity’ for PL?

- Does every derivation correspond to a valid inference?
- Does every valid inference in PL correspond to a derivation?

**How would we even show this?...**

## First Order (Predicate) Logic: Syntax and Natural Deduction<sup>1</sup>

### A Reminder of Our Plot

- I wish to provide some historical and intellectual context to the formal tools that logicians developed to study the semantics of artificial languages.
- For this reason, I'm beginning with a *purely syntactic* presentation of two key logical systems: Propositional Logic (PL) and First Order (Predicate) Logic (FOL).
- In our last notes, we covered PL. Now, we'll get a (syntactic) introduction to FOL.

## 1. A Review of First Order (Predicate) Logic (FOL): Syntax and Informal Semantics<sup>2</sup>

The system of First Order Logic (FOL) is intended to capture the inferences that depend upon:

- (i) the meaning of the so-called ‘sentential connectives’: *and*, *or*, *if...then*, and *not*
- (ii) the meaning of the quantifiers ‘every’ and ‘some’

### (1) The Vocabulary of Symbols

#### a. The Logical Constants:

(i)	$\sim$	Negation	‘It is not the case that...’
(ii)	&	Conjunction	‘and’
(iii)	$\vee$	Disjunction	‘or’ (inclusive)
(iv)	$\rightarrow$	(Material) Implication	‘if...then’
(v)	$\forall$	Universal Quantifier	‘for all...’
(vi)	$\exists$	Existential Quantifier	‘there is an...’

#### b. Syntactic Symbols: ( , )

##### a. The Non-Logical Constants (a.k.a ‘The Logical Variables’)

- (i) An infinite set of *predicate letters*: {P, Q, R, B, ... P<sub>1</sub>, P<sub>2</sub>, P<sub>3</sub>, P<sub>4</sub>, ... }
  - Each predicate letter has an associated ‘arity’ (unary, binary, etc.)
  - Each predicate letter ‘stands for’ a property or a relation
- (ii) An infinite set of *individual constants*: {a, b, c, ..., a<sub>1</sub>, a<sub>2</sub>, a<sub>3</sub>, ... }
  - The individual constants ‘stand for’ proper names (Bill, John, etc.)
- (iii) An infinite set of *variables*: {x, y, z, ..., x<sub>1</sub>, x<sub>2</sub>, x<sub>3</sub>, ... }

<sup>1</sup> These notes are based upon material in the following required reading: Gamut (1991), Chapter 3 pp. 65-83, Chapter 4 pp. 128-148; Partee *et al.* (1993), Chapter 7 pp. 135-140.

<sup>2</sup> My discussion here will assume prior familiarity with the overall system of First Order Logic. Students are referred to Partee *et al.* (1993), Chapter 7 for crucial background.

(2) **The Definition of a ‘Well-Formed Formula’ (WFF) in FOL**

The set of ‘well-formed formulae’ of PL, WFF, is the smallest set such that:

- a. If  $\varphi$  is an n-ary predicate letter and each of  $\alpha_1, \dots, \alpha_n$  is either an individual constant or a variable, then  $\varphi\alpha_1\dots\alpha_n \in \text{WFF}$
- b. If  $\varphi, \psi \in \text{WFF}$ , then
  - 1.  $\sim\varphi \in \text{WFF}$
  - 2.  $(\varphi \& \psi) \in \text{WFF}$
  - 3.  $(\varphi \vee \psi) \in \text{WFF}$
  - 4.  $(\varphi \rightarrow \psi) \in \text{WFF}$
- c. If  $\varphi \in \text{WFF}$  and  $v$  is a variable, then
  - 1.  $\forall v\varphi \in \text{WFF}$
  - 2.  $\exists v\varphi \in \text{WFF}$

Notes:

- The clause in (2a) creates the *atomic formulae* of FOL. The clause in (2c) creates the *universal formulae* and *existential formulae*.
- The set WFF includes formulae with ‘free variables’ and ‘vacuous quantification’ (defined properly later)

Hxb	(free variables)
$\exists y\forall xHxb$	(vacuous quantification)
$\exists yHxb$	(free variables and vacuous quantification)

(3) **Using FOL To Encode Sentences of English**

We can use the syntactic rules in (2) and the informal semantics in (1) to write FOL formulae that ‘encode’ certain statements of English:

- a. *Sentence:* ‘If Bill or John is leaving, then Mary and Sue aren’t happy.’  
*Encoding:*  $((Lb \vee Lj) \rightarrow (\sim Hm \& \sim Hs))$
- b. *Sentence:* ‘Every cat gave a book to Bill.’  
*Encoding:*  $\forall x(Cx \rightarrow \exists y(By \& Gxyb))$

In setting up such encodings, it is critical to supply a ‘key’, indicating what the predicate letters and individual constants ‘stand for’:

<i>Key</i>	Lx: x is leaving	b: Bill
	Hx: x is happy	j: John
	Cx: x is a cat	m: Mary
	Bx: x is a book	s: Sue
	Gxyz: x gave y to z	

**An Important Note:**

- In the key above, formulae of the form ‘Gxyz’ are interpreted so that x is the ‘subject’ of *gave*, while ‘y’ is the direct object, and ‘z’ is the indirect object.
- *Nothing forces this however.* We could just have easily had the following in our key:  
Gxyz: z gave y to x
- **Such ‘right-to-left’ readings of atomic formulae will be useful to us later, when we’re mechanically translating sentences of English into sentences of FOL...**

(4) **Key Definition: Scope**

If  $\forall v \psi$  is a subformula of  $\varphi$ , then  $\psi$  is the *scope* of (this occurrence of) ‘ $\forall v$ ’ in  $\varphi$   
If  $\exists v \psi$  is a subformula of  $\varphi$ , then  $\psi$  is the *scope* of (this occurrence of) ‘ $\exists v$ ’ in  $\varphi$

Illustration: In the formula ‘ $\exists x(Qx \ \& \ \forall y(Py \rightarrow \exists zSxyz))$ ’

- |       |                          |   |   |
|-------|--------------------------|---|---|
| (i)   | The scope of $\exists x$ | = | $(Qx \ \& \ \forall y(Py \rightarrow \exists zSxyz))$ |
| (ii)  | The scope of $\forall y$ | = | $(Py \rightarrow \exists zSxyz)$                      |
| (iii) | The scope of $\exists z$ | = | $Sxyz$  |

(5) **Key Definition: Free and Bound Variables**

- a. An occurrence of the variable  $v$  in the formula  $\varphi$  is *free in*  $\varphi$  if (i) and (ii) hold:  
(i) that occurrence of  $v$  does not occur directly to the right of either  $\exists$  or  $\forall$   
(ii) that occurrence of  $v$  is not in the scope of any occurrence of  $\exists v$  or  $\forall v$  in  $\varphi$
- b. An occurrence of the variable  $v$  is *bound by*  $\forall v$  ( $\exists v$ ) in  $\varphi$  if (i) and (ii) hold:  
(i)  $\forall v \psi$  ( $\exists v \psi$ ) is a subformula of  $\varphi$   
(ii) That occurrence of  $v$  is free in  $\psi$

Illustration: In the formula ‘ $\forall x(Px \ \& \ \exists xBx)$ ’

- |       |  |
|-------|--|
| (i)   | The occurrence of ‘ $x$ ’ in ‘ $Px$ ’ is free in ‘ $(Px \ \& \ \exists xBx)$ ’               |
| (ii)  | The occurrence of ‘ $x$ ’ in ‘ $Bx$ ’ is free in ‘ $Bx$ ’                                    |
| (iii) | The occurrence of ‘ $x$ ’ in ‘ $Bx$ ’ is <i>not</i> free in ‘ $(Px \ \& \ \exists xBx)$ ’    |
| (iv)  | The first occurrence of ‘ $x$ ’ in ‘ $(Px \ \& \ \exists xBx)$ ’ is bound by ‘ $\forall x$ ’ |
| (v)   | The occurrence of ‘ $x$ ’ in ‘ $Bx$ ’ is bound by ‘ $\exists x$ ’                            |
| (vi)  | The occurrence of ‘ $x$ ’ in ‘ $Bx$ ’ is <i>not</i> bound by ‘ $\forall x$ ’                 |

(6) **Key Definition: Sentence**

$\varphi$  is a *sentence* of FOL if (i)  $\varphi \in \text{WFF}$ , and (ii) there are no free variables in  $\varphi$

## 2. A Review of First Order Logic (FOL): Natural Deduction

### (7) Major Goal of This Section

Let's provide a *purely syntactic* characterization of 'valid inference' in the FOL notation.

- This syntactic characterization will be embodied in a *proof system* (natural deduction)
- We're going to lay out some rules – stated entirely in *syntactic terms* – for deriving formulae in FOL from other formulae.
  - As we'll see, these syntactic rules intuitively capture certain key aspects of the everyday meaning of the English logical words 'every' and 'some'

### (8) Features Inherited from PL Natural Deduction

All the following will directly imported from our system of natural deduction for PL:

- a. Definition of 'derivation'
- b. Turnstyle notation ' $\vdash$ '
- c. The rules I&, E&, Repetition, I $\rightarrow$ , E $\rightarrow$ , Iv, Ev, I~, E~, EFSQ,  $\sim\sim$

### (9) Special Feature of FOL Natural Deduction

$\varphi$  can be an assumption in an FOL derivation iff  $\varphi$  is a sentence.

*Our natural deduction system for FOL adds four new rules.*

### (10) Special Notation for Statement of Deduction Rules for FOL

If  $\varphi \in \text{WFF}$ ,  $\alpha$  is an individual constant, and  $v$  is a variable, then ' $[\alpha/v]\varphi$ ' is the formula just like  $\varphi$ , except that every *free* instance of  $v$  is replaced with an instance of  $\alpha$ :

$$[b/x](Pcx \ \& \ Dabx) = (Pcb \ \& \ Dabb)$$

### (11) The Rule of 'E-Introduction' (IE)

1.	...	
...	...	
n	$[\alpha/v]\varphi$	
...	...	
m	$\exists v\varphi$	IE n

#### Intuitive Motivation:

If we can (in English) assert for a particular thing  $\alpha$  that ' $\varphi$ ' is true of  $\alpha$ , then we can assert that there is something that ' $\varphi$ ' is true of.

(12) **The Rule of ‘ $\forall$ -Elimination’ ( $E\forall$ )**

1.	...	
...	...	
n	$\forall v \varphi$	
...	...	
m	$[\alpha/v]\varphi$	$E\forall n$

Intuitive Motivation:

If we can (in English) assert that ‘ $\varphi$ ’ is true of everything, then for any particular thing  $\alpha$ , we can assert that ‘ $\varphi$ ’ is true of  $\alpha$ ,

Illustration of  $I\exists$  and  $E\forall$ :  $\forall x Px \vdash \exists x Px$

1.	$\forall x Px$	Assumption
2.	$Pb$	$E\forall 1$
3.	$\exists x Px$	$I\exists 3$

(13) **The Rule of ‘ $\forall$ -Introduction’ ( $I\forall$ )**

Intuitive Motivation:

If we can show that ‘ $\varphi$ ’ is true of an arbitrary entity  $\alpha$  (‘arbitrary’ = we’ve not assumed anything about  $\alpha$  whatsoever), then we can assert that ‘ $\varphi$ ’ is true of *everything*.

Key Definition:

If ‘ $[\alpha/v]\varphi$ ’ appears in a derivation at line  $n$ , then  $\alpha$  is *arbitrary at line n* if (i) and (ii) hold

- (i)      $\alpha$  does not appear in any (non-dropped) assumptions in the derivation
- (ii)     $\alpha$  does not appear in  $\varphi$

The Rule:

The following is an acceptable derivation, as long as  $\alpha$  is arbitrary at line  $n$ .

1.	...	
...	...	
n	$[\alpha/v]\varphi$	
...	...	
m	$\forall v \varphi$	$I\forall n$

Illustration:  $\forall x \forall y Pxy \vdash \forall x Pxx$

1.	$\forall x \forall y Pxy$	Assumption
2.	$\forall y Pay$	$E\forall 1$
3.	$Paa$	$E\forall 2$
4.	$\forall x Pxx$	$I\forall 3$
• Note:	‘Paa’ = ‘ $[a/x]Pxx$ ’, and $a$ is arbitrary (in ‘ $[a/x]Pxx$ ’) in at line 3.	

(14) **The Rule of ‘ $\exists$ -Elimination’ ( $E\exists$ )**

Intuitive Motivation:

If we can assert (i) that ‘ $\varphi$ ’ is true of something, *and* (ii) that if ‘ $\varphi$ ’ is true of an arbitrary entity  $\alpha$ , then  $\psi$  must be true (‘arbitrary’ = we’ve not assumed anything about  $\alpha$  whatsoever), then we can assert that ‘ $\psi$ ’ is true.

Key Definition:

If  $[\alpha/v]\varphi \rightarrow \psi$  appears in a derivation at line n, then  $\alpha$  is *arbitrary at line n* if (i)-(iii):

- (i)      $\alpha$  does not appear in any (non-dropped) assumptions in the derivation
- (ii)     $\alpha$  does not appear in  $\varphi$
- (iii)    $\alpha$  does not appear  $\psi$

The Rule:

The following is an acceptable derivation, as long as  $\alpha$  is arbitrary at line n.

1.	...	
...	...	
$n_1$	$\exists v\varphi$	
...	...	
$n_2$	$[\alpha/v]\varphi \rightarrow \psi$	
...	...	
$m$	$\psi$	$E\exists n_1, n_2$

Illustration:  $\exists x \forall y Pxy \vdash \forall y \exists x Pxy$

1.	$\exists x \forall y Pxy$	Assumption
2.	$\quad \quad \quad \forall y Pby$	Assumption
3.	$\quad \quad \quad   \quad \quad Pba$	$E\forall 2$
4.	$\quad \quad \quad   \quad \quad \exists x Pxa$	$I\exists 3$
5.	$\forall y Pby \rightarrow \exists x Pxa$	$I\rightarrow$
6.	$\exists x Pxa$	$E\exists 1,5$
7.	$\forall y \exists x Pxy$	$I\forall 6$

- Note:  
 $\forall y Pby$  =  $[\bar{b}/x] \forall y Pxy$ , and b is arbitrary (in  $[\bar{b}/x] \forall y Pxy \rightarrow \exists x Pxa$ ) at line 5
- Note:     $\exists x Pxa$  =  $[\bar{a}/y] \exists x Pxy$ , and a is arbitrary (in  $[\bar{a}/y] \exists x Pxy$ ) at line 6.

Side Comment:

- The rules  $I\forall$  and  $E\exists$  add significantly to the complexity of our natural deduction system.
- However, they are crucial for system to capture *all* the valid inferences in FOL
- They also greatly complicate the proof that our natural deduction system for FOL is ‘complete’

### 3. The Power of Our Natural Deduction System for FOL

Although our system for FOL has just 15 (really, 14) relatively simple rules, it can capture a great many intuitively valid inferences!

#### (15) Derivation of Quantifier Negation, Part 1 $\sim\exists xPx \vdash \forall x\sim Px$

1.	$\sim\exists xPx$	Assumption
2.	$  Pa$	Assumption
3.	$ \exists xPx$	$\exists I$ 2
4.	$ \bot$	$E\neg 1, 3$
5.	$\sim Pa$	$I\neg$
6.	$\forall x\sim Px$	$\forall I 6$

- Note: ‘ $\sim Pa$ ’ = ‘[a/x] $\sim Px$ ’ and a is arbitrary (in ‘[a/x] $\sim Px$ ’) at line 5

#### (16) Derivation of Quantifier Negation, Part 2 $\forall x\sim Px \vdash \sim\exists xPx$

1.	$\forall x\sim Px$	Assumption
2.	$ \exists xPx$	Assumption
3.	$   Pa$	Assumption
4.	$   \sim Pa$	$E\neg 1$
5.	$   \bot$	$E\neg 3, 4$
6.	$ (Pa \rightarrow \bot)$	$I\rightarrow$
7.	$ \bot$	$E\exists 2, 6$
8.	$\sim\exists xPx$	$I\neg$

- Note: ‘ $(Pa \rightarrow \bot)$ ’ = ‘[a/x](Px  $\rightarrow$   $\bot$ )’ and a is arbitrary (in ‘[a/x](Px  $\rightarrow$   $\bot$ )’ at line 6

#### (17) The Big Question

Does our system offer a *perfect* syntactic characterization of ‘validity’ for FOL?

- Does every derivation correspond to a valid inference?
- Does every valid inference in FOL correspond to a derivation?

*How would we even show this?...*

**Problem Set on Propositional Logic and First Order Logic:  
Translation and Natural Deduction<sup>1</sup>**

**(1) Translation from English to Propositional Logic**

Please encode the following English statements as formulae in PL. Be sure to include a ‘key’ indicating what English statements each propositional letter ‘stands for’.

- a. God willing, peace will come.
- b. If it isn’t summer, then it is damp and cold, if it is evening or night.
- c. If you do not help me if I need you, I will not help you if you need me.
- d. John comes only if Peter does not come.
- e. We are going, unless it is raining.
- f. If Johnny is nice, then he will get a bicycle from Santa Clause, whether he wants one or not.

**(2) Derivations in Our Natural Deduction System for Propositional Logic**

Please provide derivations in our natural deduction system for PL establishing each of the following.

- a.  $(p \vee q) \vdash ((p \rightarrow q) \rightarrow q)$
- b.  $(p \& \sim q) \vdash \sim(p \rightarrow q)$
- c.  $(p \rightarrow \sim q) \vdash (\sim p \rightarrow q)$
- d.  $(p \& (q \vee r)) \vdash ((p \& q) \vee (p \& r))$
- e.  $((p \& q) \vee (p \& r)) \vdash (p \& (q \vee r))$

**(3) Translations from English to First Order Logic**

Please encode the following English statements as formulae in FOL. Be sure to include a ‘key’ indicating what the predicate letters and individual constants ‘stand for’.

- a. Although John and Mary love each other deeply, they make each other unhappy.
- b. It is not the case that all ambitious people are not honest.
- c. Lyn got a present from John, but she didn’t get anything from Peter.
- d. Nobody lives in Hadley who wasn’t born there.
- e. People who live in Amherst or close buy own a car.
- f. If somebody is noisy, then everybody is annoyed at him.

---

<sup>1</sup> Most of these exercises are taken from Gamut (1991), Volume 1: Chapters 2, 3 and 4

(4) **Identifying Sentences of FOL**

Please identify whether the following formulae of FOL are sentences or not.

- a.  $(\exists x Pxa \ \& \ Bx)$
- b.  $\exists x(\forall y Pxy \rightarrow Bx)$
- c.  $\forall y \sim \exists x Pxy$
- d.  $\exists x(\sim Ba \rightarrow (\sim \forall y(\sim Pxy \vee Qb) \rightarrow Cy))$
- e.  $\forall x \forall y((Pxy \ \& \ By) \rightarrow \exists w Cxw)$
- f.  $(\forall x \forall y Pyy \rightarrow Bx)$
- g.  $\forall x(\forall y Pyy \rightarrow Bx)$

(5) **Proof by Induction on the Complexity of Formula**

Please show the following by constructing a proof on the complexity of formula:

Claim: No formula of FOL begins with a variable.

(6) **Derivations in Our Natural Deduction System for First Order Logic**

Please provide derivations in our natural deduction system for FOL establishing each of the following.

- a.  $\forall x(Px \ \& \ Bx) \vdash (\forall x Px \ \& \ \forall x Bx)$
- b.  $\forall x Px \ \& \ \forall x Bx \vdash \forall x(Px \ \& \ Bx)$
- c.  $\exists x(Px \ \& \ Bx) \vdash (\exists x Px \ \& \ \exists x Bx)$
- d.  $\exists x \sim Px \vdash \sim \forall x Px$
- e.  $\sim \forall x Px \vdash \exists x \sim Px$

## Propositional Logic: Formal Semantics and Valuations<sup>1</sup>

### 1. The Motivation for Developing a Formal Semantics for Propositional Logic

#### (1) A Central Goal of (Formal) Logic, Since Aristotle

To provide a *purely syntactic* characterization of valid argumentation.

#### (2) The Means of Achieving This in Formal Logic

- a. A precisely defined formal notation for representing certain aspects of the ‘logical structure’ of an assertion.
- b. A set of *syntactically defined* rules for deriving formulas in the notation from other formulas in the notation (*e.g.*, our system of ‘Natural Deduction’).

#### (3) An Immediate Challenge for This Program

How do we know when we’ve succeeded? How do we know whether our formal system indeed produces *all and only* the inferences that are valid (in the notation)?

##### a. Entailment:

A set of formulae  $S$  **entails**  $\psi$ , if ‘whenever’ every  $\varphi \in S$  is true, so is  $\psi$

- Notation:  $S \vDash \psi$
- Note: If  $S$  entails  $\psi$ , then the inference of  $\psi$  from  $S$  is valid

##### b. Soundness:

We’ve established the **soundness** of our system, if we can show that every derivation is / corresponds to a valid inference.

- Notation: If  $S \vdash \psi$ , then  $S \vDash \psi$

##### c. Completeness:

We’ve established the **completeness** of our system, if we can show that every valid inference (in the notation) corresponds to a derivation.

- Notation: If  $S \vDash \psi$ , then  $S \vdash \psi$ ,

---

<sup>1</sup> These notes are based upon material in the following required readings: Gamut (1991), Chapter 2 pp. 44-54; Partee et al. (1993), Chapter 6 pp. 97-112.

#### (4) The Main Obstacle to Proving Soundness and Completeness

The notion of ‘entailment’ in (3a) is currently too vague and informal for precise (mathematical) argumentation.

- What, exactly, does it mean to say ‘whenever all the formulae in S are true’?

**To have a precise proof of soundness/completeness, we need rigorous explication of what it means for a formula of PL to be ‘true’.**

#### (5) Towards a Formalization of ‘Truth’ for PL

Let us begin by considering our *informal* semantics for PL, and how it can be used to determine a truth-value for a PL formula:

Example Sentence: (p & ~q)

Determining Its Truth (Informally):

a. *Step One:*

- A formula of PL isn’t true or false *in and of itself*, but only relative to an (informal) interpretation (i.e., ‘key’).
- So the first thing an informal interpretation does is it maps the primitive proposition letters to particular statements of English

p: ‘Seth lives in Northampton.’  
q: ‘Rajesh lives in Northampton.’

b. *Step Two:*

Next, to determine whether the whole formula is true, we first consider the truth of the English sentences that the proposition letters are mapped to:

‘Seth lives in Northampton’ is *true*  
‘Rajesh lives in Northampton’ is *true*

c. *Step Three:*

Finally, once we have the truth-values of the English translations, the meanings of the (English) logical connectives determine a truth-value for the whole formula.

$\sim q$  = ‘It is not the case that Rajesh lives in Northampton’ = *false*  
 $(p \ \& \ \sim q)$  = *false*

*It seems, then, that an informal interpretation of PL determines a truth-value for a PL formula by*

- *Mapping the primitive proposition letters to truth-values (indirectly, through English translations)*
- *Then, for complex formulae, the main connective determines the truth-value of the whole formula, based on the truth-values of the component formulae*

*Thus, it seems we can develop a more abstract, mathematically precise conception of ‘PL interpretation’ in the following way...*

---

## 2. A Review of Propositional Logic (PL): Formal Semantics and Valuations<sup>2</sup>

### (6) Definition of a ‘Valuation’ for Propositional Logic

A valuation  $V$  is a function from the well-formed formulae of PL to the set of truth-values  $\{1,0\}$  ( $V: \text{WFF} \rightarrow \{0,1\}$ ) such that:

- a. If  $\varphi$  is a proposition letter, then  $V(\varphi) \in \{1,0\}$  (redundant, but helpful)
- b. If  $\varphi, \psi \in \text{WFF}$ , then
 

1. $V(\neg\varphi) = 1$ iff	$V(\varphi) = 0$
2. $V((\varphi \& \psi)) = 1$ iff	$V(\varphi) = 1$ and $V(\psi) = 1$
3. $V((\varphi \vee \psi)) = 1$ iff	$V(\varphi) = 1$ or $V(\psi) = 1$ (inclusive ‘or’)
4. $V((\varphi \rightarrow \psi)) = 1$ iff	$V(\varphi) = 0$ or $V(\psi) = 1$ (inclusive ‘or’)
- c.  $V(\perp) = 0$  (This is need for the proof of soundness))

### (7) The Truth-Tables of the Logical Connectives

a.	$\varphi \parallel \neg\varphi$	b.	$\varphi \mid \psi \parallel (\varphi \& \psi)$	c.	$\varphi \mid \psi \parallel (\varphi \vee \psi)$
	1    0		1    1    1		1    1    1
	0    1		1    0    0		1    0    1
			0    1    0		0    1    1
d.	$\varphi \mid \psi \parallel (\varphi \rightarrow \psi)$		0    0    0		0    0    0
	1    1    1				
	1    0    0				
	0    1    1				
	0    0    1				

---

<sup>2</sup> My discussion here will assume prior familiarity with the formal semantics of Propositional Logic, particularly truth-tables, validity, tautology, logical-equivalence, etc. Students are referred to Partee *et al.* (1993), Chapter 6 for crucial background.

### (8) Calculating Truth-Tables for Complex Formulae

- In top row, list all the proposition letters in the formula (followed by double lines).
- Then, list all the sub-formulae, going from bottom-up (followed by single lines)
- Each row below corresponds to one of the possible valuations of the proposition letters in the sentence ( $2^n$  rows, where  $n = \text{number of proposition letters}$ )

Illustration:  $((p \ \& \ q) \rightarrow \sim r)$

p	q	r		$\sim r$	$(p \ \& \ q)$	$((p \ \& \ q) \rightarrow \sim r)$
1	1	1		0	1	0
1	1	0		1	1	1
1	0	1		0	0	1
1	0	0		1	0	1
0	1	1		0	0	1
0	1	0		1	0	1
0	0	1		0	0	1
0	0	0		1	0	1

We can use this notion of a ‘valuation’ to provide a more precise characterization of our key semantic concepts!

### (9) Tautology

- a. *Informal Notion:* S is a tautology if S is ‘necessarily true’  
 b. *Formal Notion:*  $\varphi$  is a tautology if for every valuation V,  $V(\varphi) = 1$

p		$\sim p$		$(p \vee \sim p)$
1		0		1
0		1		1

### (10) Contradiction

- a. *Informal Notion:* S is a contradiction if S is ‘necessarily false’  
 b. *Formal Notion:*  $\varphi$  is a contradiction if for every valuation V,  $V(\varphi) = 0$

p		$\sim p$		$(p \ \& \ \sim p)$
1		0		0
0		1		0

### (11) Contingent

- a. *Informal Notion:* S is contingent if S is ‘possibly true and possibly false’  
 b. *Formal Notion:*  $\varphi$  is contingent if there is a valuation V such that  $V(\varphi) = 1$ , and a valuation V' such that  $V'(\varphi) = 0$

### Key Consequences

- A formula  $\varphi$  of PL is a tautology iff  $\sim\varphi$  is a contradiction.
- A formula  $\varphi$  of PL is contingent iff  $\sim\varphi$  is contingent

## (12) Logical Equivalence

a. *Informal Notion:* S and S' are logically equivalent if S is true whenever S' is.

b. *Formal Notion:*

$\varphi$  and  $\psi$  are logically equivalent if for every valuation V,  $V(\varphi) = V(\psi)$

Illustration:  $p, \sim\sim p$

<u>p</u>	<u><math>\sim p</math></u>	<u><math>\sim\sim p</math></u>
1	0	1
0	1	0

### Key Consequences

- Formulae  $\varphi$  and  $\psi$  of PL are logically equivalent iff  $\sim\varphi$  and  $\sim\psi$  are logically equivalent.
- If  $\varphi$  and  $\psi$  are both tautologies, then  $\varphi$  and  $\psi$  are logically equivalent.
- If  $\varphi$  and  $\psi$  are both contradictions, then  $\varphi$  and  $\psi$  are logically equivalent.

Given our definition of valuation in (6), our definition of ‘logical equivalence’ in (12b) captures a wide variety of intuitive equivalences, **many of which we also derived in our proof system!**

## (13) DeMorgans Laws: $(p \vee q)$ is Logically Equivalent to $\sim(\sim p \And \sim q)$

<u>p</u>	<u>q</u>	<u><math>\sim(p \vee q)</math></u>	<u><math>\sim p</math></u>	<u><math>\sim q</math></u>	<u><math>\sim(\sim p \And \sim q)</math></u>	<u><math>\sim(\sim(p \vee q))</math></u>
1	1	1	0	0	0	1
1	0	1	0	1	0	1
0	1	1	1	0	0	1
0	0	0	1	1	1	0

## (14) Associativity of &: $((p \vee q) \vee r)$ is Logically Equivalent to $(p \vee (q \vee r))$

<u>p</u>	<u>q</u>	<u>r</u>	<u><math>\sim(p \And q)</math></u>	<u><math>\sim(p \And (q \And r))</math></u>	<u><math>\sim((p \And q) \And r)</math></u>	<u><math>\sim((p \And (q \And r)))</math></u>
1	1	1	1	1	1	1
1	1	0	1	1	1	1
1	0	1	1	1	1	1
1	0	0	1	1	0	1
0	1	1	1	1	1	1
0	1	0	1	1	1	1
0	0	1	0	1	1	1
0	0	0	0	0	0	0

### (15) Key Result: Substitution of Logically Equivalent Formulae

*Informal Statement:*

If  $\varphi$  and  $\psi$  are logically equivalent, then replacing  $\varphi$  with  $\psi$  will have no effect on the truth-value of a larger sentence.

*Formal Statement:*

Suppose that  $\varphi$  and  $\psi$  are logically equivalent, that  $\varphi$  is a subformula of  $\chi$ , and that  $\chi'$  is the formula just like  $\chi$ , except that all instances of  $\varphi$  are replaced with  $\psi$ . It follows that  $\chi$  and  $\chi'$  are logically equivalent.

Illustration:  $(p \rightarrow q)$  and  $(\sim p \rightarrow q)$ :

p	q	$\parallel (p \rightarrow q)$	$\sim p$	$\sim \sim p$	$(\sim \sim p \rightarrow q)$
1	1	1	0	1	1
1	0	0	0	1	0
0	1	1	1	0	1
0	0	1	1	0	1

### (16) Key Result: Dropping Operators from Our System

- Informally speaking, what (15) shows is that if  $\varphi$  and  $\psi$  are logically equivalent, then anything we can ‘express’ in PL with  $\varphi$ , we can also ‘express’ with  $\psi$
- Now, recall the following key equivalences (proofs left to the student):
  - (i)  $(\varphi \rightarrow \psi)$  is logically equivalent to  $(\sim \varphi \vee \psi)$
  - (ii)  $(\varphi \vee \psi)$  is logically equivalent to  $\sim(\sim \varphi \& \sim \psi)$
  - (iii)  $(\varphi \& \psi)$  is logically equivalent to  $\sim(\sim \varphi \vee \sim \psi)$
- It follows from (i) that anything we can ‘express’ in PL with ‘ $\rightarrow$ ’, we can also express with ‘ $\sim$ ’ and ‘ $\vee$ ’.
- It follows from (ii) that anything we can ‘express’ in PL with ‘ $\vee$ ’, we can also express with ‘ $\sim$ ’ and ‘ $\&$ ’.
- It follows from (iii) that anything we can ‘express’ in PL with ‘ $\&$ ’ we can also express with ‘ $\sim$ ’ and ‘ $\&$ ’.
- **Consequently, we can drop ‘ $\rightarrow$ ’ from our PL system (leaving ‘ $\sim$ ’, ‘ $\&$ ’ ‘ $\vee$ ’), and still have an equivalently ‘expressive’ system**
- **Consequently, we can drop ‘ $\vee$ ’ from our system (leaving ‘ $\&$ ’ and ‘ $\sim$ ’), and still have an equivalently ‘expressive’ system**
- **Consequently, we can drop ‘ $\&$ ’ from our system (leaving ‘ $\vee$ ’ and ‘ $\sim$ ’), and still have an equivalently ‘expressive’ system**

(17) **Key Result: Defining Operators in Terms of Other Operators**

Given the result in (16), we could view the operators ' $\rightarrow$ ' and ' $\vee$ ' (or ' $\rightarrow$ ' and '&') not as primitive operators, but as *special abbreviations* for more complex formulae

- $(\varphi \rightarrow \psi)$  is 'shorthand' for  $(\sim\varphi \vee \psi)$
- $(\varphi \vee \psi)$  is 'shorthand' for  $\sim(\sim\varphi \& \sim\psi)$

We'll make use of this commonplace idea in (17) later on, as it will greatly simplify the definitions of the logical languages we use for the semantic analysis of English...

*Finally, the notion of 'valuation' provides us with a more rigorous definition of 'entailment'...*

(18) **Preliminary Definition**

If  $S$  is a set of formulae, then  $V$  is a **valuation for  $S$**  if for every  $\varphi \in S$ ,  $V(\varphi) = 1$

(19) **Entailment**

- a. *Informal Notation:*

A set of formulae  $S$  **entails**  $\psi$ , if 'whenever' every  $\varphi \in S$  is true, so is  $\psi$

- b. *Formal Notion:*

A set of formulae  $S$  **entails**  $\psi$ , if every valuation  $V$  of  $S$  is such that  $V(\psi) = 1$

(20) **Key Consequence: Entailment and Tautology**

Let  $S$  be a finite set of formulae  $\{ \varphi_1, \dots, \varphi_n \}$ .  $S$  entails  $\psi$  iff  $((\varphi_1 \& \dots \& \varphi_n) \rightarrow \psi)$  is a tautology.

*Proof:* Suppose that  $S$  entails  $\psi$ . Thus, for any valuation  $V$ , if  $V((\varphi_1 \& \dots \& \varphi_n)) = 1$  then  $V$  is a valuation for  $S$ , and so  $V(\psi) = 1$ . Thus, for any valuation  $V$ , if  $V((\varphi_1 \& \dots \& \varphi_n)) = 1$ , then  $V(\psi) \neq 0$ , and so  $V(((\varphi_1 \& \dots \& \varphi_n) \rightarrow \psi)) = 1$ .

Suppose that  $((\varphi_1 \& \dots \& \varphi_n) \rightarrow \psi)$  is a tautology. Now suppose that  $V$  is a valuation of  $S$ . It follows that  $V((\varphi_1 \& \dots \& \varphi_n)) = 1$ . Moreover, since  $((\varphi_1 \& \dots \& \varphi_n) \rightarrow \psi)$  is a tautology, it follows that  $V(((\varphi_1 \& \dots \& \varphi_n) \rightarrow \psi)) = 1$ . Consequently, it must be the case that  $V(\psi) \neq 0$ , and so  $V(\psi) = 1$ .

(21) **Major Consequence: Computability of Entailment for PL**

One can 'effectively' compute whether a finite set of PL formulae  $S$  entail a formula  $\psi$

- Just compute the truth-table for  $(S \rightarrow \psi)$  and check whether it's a tautology!

Now that we have this notion of ‘valuation’, the key issues in (3) become much more tractable!

(22) **Soundness of PL**

- If  $S \vdash \psi$ , then  $S \vDash \psi$
- If  $S \vdash \psi$ , then if  $V$  is a valuation for  $S$ ,  $V(\psi) = T$

(23) **Completeness of PL**

- If  $S \vDash \psi$ , then  $S \vdash \psi$
- If every valuation  $V$  of  $S$  is also a valuation of  $\psi$ , then  $S \vdash \psi$

*In the next set of notes, we'll see how (22) and (23) can be rigorously proved!*

## First Order Logic: Formal Semantics and Models<sup>1</sup>

### (1) Goal for These Notes

Just as we did for PL, we need a mathematically rigorous explication of what it is for a sentence of FOL to be ‘true’.

- This will give us an appropriately precise definition of ‘entailment’, which will ultimately allow us to prove that FOL is sound and complete...

### (2) Towards a Formalization of ‘Truth’ for FOL

Let’s begin by considering sentences of FOL *without quantifiers*, and how an informal semantics (‘key’) determines a truth-value for them.

Example Sentence: (Fa & ~Pbc)

Determining Its Truth (Informally)

a. *Step One: Consult the Key*

A ‘key’ for an FOL encoding maps individual constants of FOL to proper names, and predicate letters of FOL to properties/relations.

Fx: x is French

a: Angelika Kratzer

Pxy: x is older than y

b: Seth Cable

c: Rajesh Bhatt

b. *Step Two: Consult the Facts*

Next, we consider whether the properties/relations denoted by the English expressions actually hold of the entities named by the English names.

- Angelika Kratzer is not French.
- Seth Cable is not older than Rajesh Bhatt

**This determines the truth-value of the atomic sentences of FOL.**

c. *Step Three:*

Finally, once we have the truth-values of the atomic sentences, the meanings of the (English) logical connectives determine a truth-value for the whole formula.

$$\begin{array}{llll} \sim Pbc & = & \text{‘It is not the case that Seth is older than Rajesh’} & = \\ Fa & = & \text{‘Angelika Kratzer is French’} & = \\ (Fa \& \sim Pbc) & = & \text{false} & \end{array} \quad \begin{array}{ll} True \\ False \end{array}$$

<sup>1</sup> These notes are based upon material in the following required readings: Gamut (1991), Chapter 3 pp. 87-101; Partee *et al.* (1993), Chapter 7 pp. 140-152, Chapter 13 pp. 321-331.

*It seems, then, that an informal interpretation of FOL determines a truth-value for an FOL sentence (without quantification) by*

- *Mapping the individual constants to entities (indirectly, through English translations)*
- *Mapping the predicate letter to some set (property) or set of n-tuples (relation)*
- *For atomic sentences, the sentence is true if the individual(s) named by the constant are ‘in’ the set or relation denoted by the predicate letter.*
- *For complex sentences, the main connective determines the truth-value of the whole formula, based on the truth-values of the component formulae*

*Thus, it seems we can develop a more abstract, mathematically precise conception of ‘FOL interpretation’ in the following way...*

---

---

## 2. A Review of First Order Logic (FOL): Formal Semantics and Models<sup>2</sup>

### (7) Definition of a ‘Model’ for First Order Logic

A model  $\mathcal{M}$  is a pair  $\langle D, I \rangle$  consisting of:

- a non-empty set  $D$ , called the ‘domain of  $\mathcal{M}$
- A function  $I$ , whose domain is the individual constants and predicate letters, and whose range satisfies the following conditions:
  - If  $\alpha$  is an individual constant, then  $I(\alpha) \in D$
  - If  $\Phi$  is an  $n$ -ary predicate letter, then  $I(\Phi) \subseteq D^n$

Note:

- If  $\Phi$  is a unary predicate letter, then  $I(\Phi) \subseteq D^1 = D$  (a subset of  $D$ )
- If  $\Phi$  is a binary predicate letter, then  $I(\Phi) \subseteq D^2 = D \times D$  (a set of pairs from  $D$ )
- If  $\Phi$  is a ternary predicate letter, then  $I(\Phi) \subseteq D^3 = D \times D \times D$  (a set of triples from  $D$ ), etc.

### (8) Illustration:

The following is a model of FOL  $\langle \{ \text{Angelika, Seth, Rajesh} \}, I \rangle$ , where  $I$  consists of at least the following mappings:

$$I(a) = \text{Angelika} \quad I(b) = \text{Seth} \quad I(c) = \text{Rajesh}$$

$$I(F) = \emptyset \quad I(P) = \{ \langle \text{Angelika, Rajesh} \rangle, \langle \text{Angelika, Seth} \rangle, \langle \text{Rajesh, Seth} \rangle \}$$

---

<sup>2</sup> My discussion here will assume prior familiarity with the formal semantics of First Order Logic. Students are referred to Partee *et al.* (1993), Chapter 7 for crucial background.

With this notion of ‘model’ in place, we can introduce the key formal semantic notion below:

**(9) Valuation of FOL, Relative to a Model (To Be Revised)**

Let  $\mathcal{M}$  be a model  $\langle D, I \rangle$ . Then the ‘valuation based on  $M$ ’ ( $V_M$ ) is a function whose domain is the set of FOL sentences (without quantifiers), whose range is  $\{0,1\}$ , and which satisfies the conditions below:

- (i) If  $\varphi = \Phi \alpha_1 \dots \alpha_n$ , then  $V_M(\varphi) = 1$  iff  $\langle I(\alpha_1) \dots I(\alpha_n) \rangle \in I(\Phi)$
- (ii) If  $\varphi = \neg \psi$ , then  $V_M(\varphi) = 1$  iff  $V_M(\psi) = 0$
- (iii) If  $\varphi = (\psi \ \& \ \chi)$ , then  $V_M(\varphi) = 1$  iff  $V_M(\psi) = 1$  and  $V_M(\chi) = 1$
- (iv) If  $\varphi = (\psi \ \vee \ \chi)$ , then  $V_M(\varphi) = 1$  iff  $V_M(\psi) = 1$  or  $V_M(\chi) = 1$
- (v) If  $\varphi = (\psi \rightarrow \chi)$ , then  $V_M(\varphi) = 1$  iff  $V_M(\psi) = 0$  or  $V_M(\chi) = 1$

**(10) Illustration**

Let  $\mathcal{M}$  be the model (partially) defined in (8). From (9), it follows that

- $V_M(Fa \ \& \ \neg Pbc) = 1$  iff (by 9iii)
- $V_M(Fa) = 1$  and  $V_M(\neg Pbc) = 1$  iff (by 9ii)
- $V_M(Fa) = 1$  and  $V_M(Pbc) = 0$  iff (by 9i)
- $I(a) \in I(F)$  and  $\langle I(b), I(c) \rangle \notin I(P)$  iff (by definition of  $I$  in (8))
- $Angelika \in \emptyset$  and  $\langle Seth, Rajesh \rangle \notin \{ \langle Angelika, Rajesh \rangle, \langle Angelika, Seth \rangle, \langle Rajesh, Seth \rangle \}$

Thus, we can calculate that  $V_M(Fa \ \& \ \neg Pbc) = 0$

In this way, the definition of ‘model’ in (8) and ‘valuation’ in (9) mirrors the way our earlier ‘informal interpretation’ (key) maps the formula ‘ $(Fa \ \& \ \neg Pbc)$ ’ to the truth-value *false*

**PROBLEM:** We forgot about quantifiers! How do we add them into the picture?

(11) **Key Problem: The Semantics of Formulae with Free Variables**

Consider an FOL sentence like ‘ $\exists x Px$ ’.

- This is a complex formula made up of the quantifier ‘ $\exists x$ ’ and the atomic formula ‘ $Px$ ’
- Moreover, we’re going to want this formula to end up being entailed by ‘ $Pa$ ’.
- Thus, we’re going to want the truth-value of this formula to somehow be determined by *some kind of semantic value* that the formula ‘ $Px$ ’ has (and for that value to ‘connected’ with the truth of  $Pa$  in some fashion...)
- **BUT HOW IN GOD’S NAME DO WE DO THAT???**

(12) **A Naive Intuition, to Get Us Started**

- In English, “Something is red” is true *iff* there is a particular thing  $x$  such that we could ‘point to it’ and truthfully say ‘THAT is red’.
- In English, “Everything is red” is true *iff* for any thing in the world  $x$ , you could ‘point to it’, and truthfully say ‘THAT is red’.

(13) **Building From That Naïve Intuition**

- $\exists x\varphi$  is true *iff* there is a thing  $\alpha$  such that  $\varphi$  is true when  $x$  ‘picks out’  $\alpha$
- $\forall x\varphi$  is true *iff* for any thing  $\alpha$ ,  $\varphi$  is true when  $x$  ‘picks out’  $\alpha$

*But what do we mean by ‘when  $x$  picks out  $\alpha$ ’?!?*

*Well, this will be familiar to the semanticists...*

(14) **Variable Assignment**

Let  $\mathcal{M}$  be a model  $\langle D, I \rangle$ . Then  $g$  is a *variable assignment* (*based on  $\mathcal{M}$* ) if:

- $g$  is a function whose domain is the set of all the variables in FOL,
- and whose range is  $D$

Note:  $g$  needn’t be an injection; it could map two different variables to the same  $\alpha \in D$ !

(15) **Notation**

Let  $\mathcal{M}$  be a model  $\langle D, I \rangle$ , and let  $g$  be a variable assignment based on  $\mathcal{M}$ , and let  $\alpha \in D$ , and let  $v$  be a variable of FOL.

$g(v/\alpha)$  is the variable assignment exactly like  $g$  except (at most) that it maps  $v$  to  $\alpha$

Note: The following follow from the definition in (15)

- $g(x/a)(y/b)$  is just like  $g$  except (at most) that it maps  $x$  to  $a$  and  $y$  to  $b$ .
- $g(x/a)(x/b) = g(x/b)$

(16) **The Term ‘Term’**

$t$  is a ‘term (of FOL)’ iff  $t$  is either a variable or an individual constant

(17) **Some More Notation**

Let  $t$  be a term,  $\mathcal{M}$  be a model  $\langle D, I \rangle$  and  $g$  be a variable assignment based on  $\mathcal{M}$ . Then the interpretation of  $t$  relative to  $\mathcal{M}$  and  $g$   $[[t]]^{M,g}$  is defined as follows:

- a. If  $t$  is an individual constant, then  $[[t]]^{M,g} = I(t)$
- b. If  $t$  is a variable, then  $[[t]]^{M,g} = g(t)$

*With these notions in place, we can revise our notion of ‘valuation’ in (9) so that it now can map every formula (without quantifiers) to truth values, even those with free variables!*

(18) **Valuation of FOL, Relative to a Model and a Variable Assignment (To Be Revised)**

Let  $\mathcal{M}$  be a model  $\langle D, I \rangle$  and  $g$  be a variable assignment (based on  $\mathcal{M}$ ). Then the ‘valuation based on  $M$  and  $g$ ’ ( $V_{M,g}$ ) is a function whose domain is the set of FOL formulae (without quantifiers), whose range is  $\{0,1\}$ , and which satisfies the conditions below:

- (i) If  $\varphi = \Phi t_1 \dots t_n$ , then  $V_{M,g}(\varphi) = 1$  iff  $\langle [[t_1]]^{M,g}, \dots, [[t_n]]^{M,g} \rangle \in I(\Phi)$
- (ii) If  $\varphi = \neg\psi$ , then  $V_{M,g}(\varphi) = 1$  iff  $V_{M,g}(\psi) = 0$
- (iii) If  $\varphi = (\psi \ \& \ \chi)$ , then  $V_{M,g}(\varphi) = 1$  iff  $V_{M,g}(\psi) = 1$  and  $V_{M,g}(\chi) = 1$
- (iv) If  $\varphi = (\psi \vee \chi)$ , then  $V_{M,g}(\varphi) = 1$  iff  $V_{M,g}(\psi) = 1$  or  $V_{M,g}(\chi) = 1$
- (v) If  $\varphi = (\psi \rightarrow \chi)$ , then  $V_{M,g}(\varphi) = 1$  iff  $V_{M,g}(\psi) = 0$  or  $V_{M,g}(\chi) = 1$

(19) **Illustration**

Let  $\mathcal{M}$  be the model (partially) defined in (8) and let  $g$  be the following variable assignment:  $\{\langle x, \text{Rajesh} \rangle, \langle y, \text{Seth} \rangle, \langle z, \text{Angelika} \rangle\}$ . From (18), it follows that

- $V_{M,g}(\neg P xc) = 1$  iff (by 18ii)
- $V_{M,g}(P xc) = 0$  iff (by 18i)
- $\langle [[x]]^{M,g}, [[c]]^{M,g} \rangle \notin I(P)$  iff (by definition in (17))
- $\langle g(x), I(c) \rangle \notin I(P)$  iff (by definition of  $g$  and  $I$ )
- $\langle \text{Rajesh}, \text{Rajesh} \rangle \notin \{ \langle \text{Angelika}, \text{Rajesh} \rangle, \langle \text{Angelika}, \text{Seth} \rangle, \langle \text{Rajesh}, \text{Seth} \rangle \}$

Thus, we can calculate that  $V_{M,g}(\neg P xc) = 1$

*Our notion of ‘interpretation (valuation) with respect to a variable assignment’ explicates what we mean by ‘when a variable picks out α’*

- ‘Φ is true when x ‘picks out’ α’       $\approx$        $V_{M,g}(\Phi) = 1$  if  $g(x) = \alpha$

*With this in mind we can extend our definition in (18) to include quantifiers!*

## (20) Extending (18) to Include Quantifiers

- a. What We Ultimately Want: Two equations whose left hand side looks like:

- (vi) If  $\varphi = \exists v\psi$ , then  $V_{M,g}(\varphi) = 1$  iff ...  
 (vii) If  $\varphi = \forall v\psi$ , then  $V_{M,g}(\varphi) = 1$  iff ...

- b. Recall Our Naïve Intuition:

- (i)  $\exists x\psi$  is true iff there is a thing  $\alpha$  such that  $\psi$  is true when  $x$  ‘picks out’  $\alpha$   
 (ii)  $\forall x\psi$  is true iff for any thing  $\alpha$ ,  $\psi$  is true when  $x$  ‘picks out’  $\alpha$

- c. Spelling Out the Naïve Intuition With Variable Assignments

- (vi) If  $\varphi = \exists v\psi$ , then  $V_{M,g}(\varphi) = 1$  iff **there is an  $a \in D$  such that  $V_{M,g(v/a)}(\psi) = 1$**   
 (vii) If  $\varphi = \forall v\psi$ , then  $V_{M,g}(\varphi) = 1$  iff **for every  $a \in D$ ,  $V_{M,g(v/a)}(\psi) = 1$**
- In (vi) and (vii), we interpret  $\psi$  relative to the variable assignment  $g(v/a)$ ; thus, we interpret  $\psi$  with ‘ $v$  picking out  $a$ ’

## (21) Valuation of FOL, Relative to a Model and a Variable Assignment (Final Version)

Let  $\mathcal{M}$  be a model  $\langle D, I \rangle$  and  $g$  be a variable assignment (based on  $\mathcal{M}$ ). Then the ‘valuation based on  $M$  and  $g$ ’ ( $V_{M,g}$ ) is a function whose domain is the set of FOL formulae, whose range is  $\{0,1\}$ , and which satisfies the conditions below:

- (i) If  $\varphi = \Phi t_1 \dots t_n$ , then  $V_{M,g}(\varphi) = 1$  iff  $\langle [t_1], \dots, [t_n] \rangle^{M,g} \in I(\Phi)$   
 (ii) If  $\varphi = \neg\psi$ , then  $V_{M,g}(\varphi) = 1$  iff  $V_{M,g}(\psi) = 0$   
 (iii) If  $\varphi = (\psi \& \chi)$ , then  $V_{M,g}(\varphi) = 1$  iff  $V_{M,g}(\psi) = 1$  and  $V_{M,g}(\chi) = 1$   
 (iv) If  $\varphi = (\psi \vee \chi)$ , then  $V_{M,g}(\varphi) = 1$  iff  $V_{M,g}(\psi) = 1$  or  $V_{M,g}(\chi) = 1$   
 (v) If  $\varphi = (\psi \rightarrow \chi)$ , then  $V_{M,g}(\varphi) = 1$  iff  $V_{M,g}(\psi) = 0$  or  $V_{M,g}(\chi) = 1$   
 (vi) **If  $\varphi = \exists v\psi$ , then  $V_{M,g}(\varphi) = 1$  iff there is an  $a \in D$  such that  $V_{M,g(v/a)}(\psi) = 1$**   
 (vii) **If  $\varphi = \forall v\psi$ , then  $V_{M,g}(\varphi) = 1$  iff for every  $a \in D$ ,  $V_{M,g(v/a)}(\psi) = 1$**

(22) **Illustration**<sup>3</sup>

Let  $\mathcal{M}$  be a model  $\langle \{\text{Dave, Joe}\}, I \rangle$ , where  $I(L) = \{\langle \text{Dave, Joe} \rangle, \langle \text{Joe, Dave} \rangle\}$ . Let  $g$  be a variable assignment such that  $g(x) = \text{Dave}$  and  $g(y) = \text{Joe}$ .

a. Interpreting ' $\exists y Lxy$ ' with respect to  $\mathcal{M}$  and  $g$

1.  $V_{M,g}(\exists y Lxy) = 1$  iff (by (21vi))
2. There is an  $a \in D$  s.t.  $V_{M,g(y/a)}(Lxy) = 1$  iff (by (21i))
3. There is an  $a \in D$  s.t.  $\langle [[x]]^{M,g(y/a)}, [[y]]^{M,g(y/a)} \rangle \in I(L)$  iff
4. There is an  $a \in D$  s.t.  $\langle g(y/a)(x), g(y/a)(y) \rangle \in I(L)$  iff (by def. of  $g$  and  $I$ )
5. There is an  $a \in D$  s.t.  $\langle \text{Dave}, a \rangle \in \{\langle \text{Dave, Joe} \rangle, \langle \text{Joe, Dave} \rangle\}$

**Thus, we can calculate that  $V_{M,g}(\exists y Lxy) = 1$**

b. Interpreting ' $\forall x \exists y Lxy$ ' with respect to  $\mathcal{M}$  and  $g$

1.  $V_{M,g}(\forall x \exists y Lxy) = 1$  iff (by (21vii))
2. For every  $a \in D$ ,  $V_{M,g(x/a)}(\exists y Lxy) = 1$  iff (by (21vi))
3. For every  $a \in D$ , there is an  $a' \in D$  s.t.  $V_{M,g(x/a)(y/a')}(\exists y Lxy) = 1$  iff (by (21i))
4. For every  $a \in D$ , there is an  $a' \in D$  s.t.  
 $\langle [[x]]^{M,g(x/a)(y/a')}, [[y]]^{M,g(x/a)(y/a')} \rangle \in I(L)$  iff
5. For every  $a \in D$ , there is an  $a' \in D$  s.t.  
 $\langle g(x/a)(y/a')(x), g(x/a)(y/a')(y) \rangle \in I(L)$  iff (by def. of  $g$  and  $I$ )
6. For every  $a \in D (\{\text{Dave, Joe}\})$ , there is an  $a' \in D (\{\text{Dave, Joe}\})$  s.t.  
 $\langle a, a' \rangle \in \{\langle \text{Dave, Joe} \rangle, \langle \text{Joe, Dave} \rangle\}$

**Thus, we can calculate that  $V_{M,g}(\forall x \exists y Lxy) = 1$**

- We've found a viable way of using *models* to assign truth-values to formulae of FOL
  - So, we've also found a way of using models to assign truth-values to FOL sentences
    - So, *models seem like an excellent characterization of 'interpretation' for FOL*

<sup>3</sup> For another helpful illustration of the key definitions in (21), see Partee *et al.* (1993), Chapter 13 pp. 326-327.

### 3. Some Important Consequences and Notations

#### (23) Key Fact: Sentences of FOL and Variable Assignments

Let  $\mathcal{M}$  be any model and  $g, h$  be variable assignments based on  $\mathcal{M}$ . If  $\varphi$  is a sentence of FOL, then  $V_{\mathcal{M},g}(\varphi) = V_{\mathcal{M},h}(\varphi)$

- That is, the truth-value of a *sentence* relative to a model and a variable assignment doesn't depend upon the variable assignment at all (since there are no free variables)
  - Notice how in (22b), we don't ever actually calculate the value of  $g$  for any variable.
  - Thus, even if we replaced  $g$  with some other variable assignment  $h$  in (22b), we'd still get the same result!...
- Consequently, if a sentence  $\varphi$  is true with respect to a model  $\mathcal{M}$  and a variable assignment  $g$ , **then it's true with respect to  $\mathcal{M}$  and any variable assignment  $h$**

*So, even though we need to refer to a variable assignment to calculate the truth-value of a sentence, the sentence's truth-value doesn't depend upon the assignment we pick...*

*For this reason, we can introduce the following terminology:*

#### (24) Key Terminology: A Model of an FOL Sentence (Set of Sentences)

Let  $\mathcal{M}$  be a model,  $\varphi$  be a sentence of FOL, and  $S$  be a set of FOL sentences.

- a.  $\varphi$  is *true in  $\mathcal{M}$*       iff for any variable assignment  $g$ ,  $V_{\mathcal{M},g}(\varphi) = 1$
- b.  $\mathcal{M}$  is a model for  $\varphi$       iff  $\varphi$  is *true in  $\mathcal{M}$*
- c.  $\mathcal{M}$  is a model for  $S$       iff for all  $\varphi \in S$ ,  $\mathcal{M}$  is a model for  $\varphi$

*Also, if we want to – and some do – we can eliminate direct reference to  $V_{\mathcal{M},g}$  in our semantics for FOL, by extending the notation ' $[\cdot]$ ' $^{M,g}$ ', defined in (17)*

(25) **Interpretation With Respect to a Model**

Let  $\mathcal{M}$  be a model  $\langle D, I \rangle$  and  $g$  be a variable assignment based on  $\mathcal{M}$ . Then the interpretation (a.k.a. denotation) of an ‘expression’ of FOL relative to  $M$  and  $g$   $[[\cdot]]^{M,g}$  is defined as follows:

- a. If  $v$  is a variable, then  $[[v]]^{M,g} = g(v)$
- b. If  $\alpha$  is an individual constant, then  $[[\alpha]]^{M,g} = I(\alpha)$
- c. If  $\Phi$  is a predicate letter, then  $[[\Phi]]^{M,g} = I(\Phi)$
- d. If  $\varphi = \Phi t_1 \dots t_n$ , then  $[[\varphi]]^{M,g} = 1$  iff  $\langle [[t_1]]^{M,g}, \dots, [[t_n]]^{M,g} \rangle \in [[\Phi]]^{M,g}$
- e. If  $\varphi = \sim\psi$ , then  $[[\varphi]]^{M,g} = 1$  iff  $[[\psi]]^{M,g} = 0$
- f. If  $\varphi = (\psi \& \chi)$ , then  $[[\varphi]]^{M,g} = 1$  iff  $[[\psi]]^{M,g} = 1$  and  $[[\chi]]^{M,g} = 1$
- g. If  $\varphi = (\psi \vee \chi)$ , then  $[[\varphi]]^{M,g} = 1$  iff  $[[\psi]]^{M,g} = 1$  or  $[[\chi]]^{M,g} = 1$
- h. If  $\varphi = (\psi \rightarrow \chi)$ , then  $[[\varphi]]^{M,g} = 1$  iff  $[[\psi]]^{M,g} = 0$  or  $[[\chi]]^{M,g} = 1$
- i. If  $\varphi = \exists v\psi$ , then  $[[\varphi]]^{M,g} = 1$  iff there is an  $a \in D$  such that  $[[\psi]]^{M,g(v/a)} = 1$
- j. If  $\varphi = \forall v\psi$ , then  $[[\varphi]]^{M,g} = 1$  iff for all  $a \in D$ ,  $[[\psi]]^{M,g(v/a)} = 1$

(26) **Notation**

Let  $\varphi$  be a sentence of FOL and  $\mathcal{M}$  be a model. The interpretation (a.k.a. denotation) of  $\varphi$  with respect to  $M$ ,  $[[\varphi]]^M$ , is  $[[\varphi]]^{M,g}$  for an arbitrary variable assignment  $g$ .

Consequences:

- $\varphi$  is true in  $\mathcal{M}$  iff  $[[\varphi]]^M = 1$  iff  $\mathcal{M}$  is a model for  $\varphi$
- $\mathcal{M}$  is a model for  $S$  iff for all  $\varphi \in S$ ,  $[[\varphi]]^M = 1$

**4. Using Models to Define Semantic Concepts for FOL**

*With our formal characterization of ‘truth’ for FOL, we can give precise definitions to the concepts of ‘tautology’, ‘contradiction’, ‘logical equivalence’ and ‘entailment’....*

*These definitions are going to be restricted to sentences of FOL...*

(27) **Tautology**

Let  $\varphi$  be a sentence of FOL.  $\varphi$  is a tautology iff for every model  $\mathcal{M}$   $[[\varphi]]^{\mathcal{M}} = 1$

Illustration:  $\forall x(Px \vee \neg Px)$

Proof That It's a Tautology

Let  $\mathcal{M}$  be any model  $\langle D, I \rangle$  and  $g$  be any variable assignment (based on  $\mathcal{M}$ ).

- Suppose that  $[[\forall x(Px \vee \neg Px)]]^{M,g} = 0$
- Then it's not the case that for all  $a \in D$ ,  $[[Px \vee \neg Px]]^{M,g(x/a)} = 1$
- Then there is some  $a \in D$  such that  $[[Px \vee \neg Px]]^{M,g(x/a)} = 0$ .
- Then there is some  $a \in D$  such that  $[[Px]]^{M,g(x/a)} = 0$  and  $[[\neg Px]]^{M,g(x/a)} = 0$
- Then there is some  $a \in D$  such that  $[[Px]]^{M,g(x/a)} = 0$  and  $[[Px]]^{M,g(x/a)} = 1$
- Then there is some  $a \in D$  such that  $g(x/a)(x) \notin I(P)$  and  $g(x/a)(x) \in I(P)$
- Then there is some  $a \in D$  such that  $a \notin I(P)$  and  $a \in I(P)$  **CONTRADICTION**

**Important Note:**

It's common for logicians to refer to FOL sentences true in every model as *universally valid*.

- The term 'tautology' is often restricted to FOL formulae that can be obtained by taking a tautology of PL and replacing the propositional letters with FOL sentences.

(28) **Contradiction**

Let  $\varphi$  be a sentence of FOL.  $\varphi$  is a contradiction iff for every model  $\mathcal{M}$   $[[\varphi]]^{\mathcal{M}} = 0$

Illustration:  $\exists x(Px \wedge \neg Px)$

Proof That It's a Contradiction

Let  $\mathcal{M}$  be any model  $\langle D, I \rangle$  and  $g$  be any variable assignment (based on  $\mathcal{M}$ ).

- Suppose that  $[[\exists x(Px \wedge \neg Px)]]^{M,g} = 1$
- Then there is some  $a \in D$  such that  $[[Px \wedge \neg Px]]^{M,g(x/a)} = 1$ .
- Then there is some  $a \in D$  such that  $[[Px]]^{M,g(x/a)} = 1$  and  $[[\neg Px]]^{M,g(x/a)} = 1$
- Then there is some  $a \in D$  such that  $[[Px]]^{M,g(x/a)} = 1$  and  $[[Px]]^{M,g(x/a)} = 0$
- Then there is some  $a \in D$  such that  $g(x/a)(x) \in I(P)$  and  $g(x/a)(x) \notin I(P)$
- Then there is some  $a \in D$  such that  $a \in I(P)$  and  $a \notin I(P)$  **CONTRADICTION**

(29) **Contingent**

Let  $\varphi$  be a sentence of FOL.  $\varphi$  is contingent iff there are models  $\mathcal{M}$  and  $\mathcal{M}'$  such that  $[[\varphi]]^{\mathcal{M}} = 1$  and  $[[\varphi]]^{\mathcal{M}'} = 0$ .

Illustration:  $\exists x(Px \wedge Qx)$  [proof trivial]

(30) **Consequences**

- $\varphi$  is a tautology (universally valid) iff  $\sim\varphi$  is a contradiction
- $\varphi$  is contingent iff  $\sim\varphi$  is contingent

(31) **Key Observation**

- Note that in showing that a given sentence of FOL is a tautology or contradiction, we need to use some (modest) mathematical ingenuity.
  - Unlike PL, we don't just 'mechanically calculate' it out (via truth-tables)
- **This is no accident. A very deep and interesting theorem of computability theory is that there can be no purely mechanical, algorithmic procedure for calculating whether an FOL sentence is a tautology or not.**
- Similarly, there is no purely mechanical procedure for calculating whether or not one FOL sentence 'entails' another (defined below) or whether two sentences of FOL are 'logically equivalent' (defined below).
  - Unlike PL, 'you gotta use your noodle...''

(32) **Logical Equivalence**

Let  $\varphi, \psi$  be sentences of FOL.  $\varphi$  and  $\psi$  are logically equivalent if for every model  $\mathcal{M}$

$$[[\varphi]]^{\mathcal{M}} = [[\psi]]^{\mathcal{M}}$$

Illustration:  $\forall x \sim P x$  and  $\sim \exists x P x$

Proof of Logical Equivalence:

Let  $\mathcal{M}$  be any model  $\langle D, I \rangle$  and  $g$  be any variable assignment (based on  $\mathcal{M}$ ).

- $[[\forall x \sim P x]]^{M,g} = 1$  iff
- For all  $a \in D$ ,  $[[\sim P x]]^{M,g(x/a)} = 1$  iff
- For all  $a \in D$ ,  $[[P x]]^{M,g(x/a)} = 0$  iff
- For all  $a \in D$ ,  $a \notin I(P)$  iff
- **It is not the case that there is an  $a \in D$  s.t.  $a \in I(P)$**  iff
- It is not the case that there is an  $a \in D$  s.t.  $[[P x]]^{M,g(x/a)} = 1$  iff
- It is not the case that  $[[\exists x P x]]^{M,g} = 1$  iff
- $[[\exists x P x]]^{M,g} = 0$  iff
- $[[\sim \exists x P x]]^{M,g} = 1$  iff

(33) **Consequences**

- Any two tautologies (universal validities) are logically equivalent
- Any two contradictions are logically equivalent.
- If  $\varphi, \psi$  are logically equivalent, then so are  $\sim\varphi, \sim\psi$

(34) **Consequence**

The following two formulae are logically equivalent:  $\exists xPx$  and  $\sim\forall x\sim Px$

- $\forall x\sim Px$  and  $\sim\exists xPx$  are logically equivalent (proof in (32))
- $\sim\forall x\sim Px$  and  $\sim\sim\exists xPx$  are logically equivalent (consequences in (33))
- $\sim\forall x\sim Px$  and  $\exists xPx$  are logically equivalent (double negation)

(35) **Key Result: Substitution of Logically Equivalent Formulae**

*Informal Statement:*

If  $\varphi$  and  $\psi$  are logically equivalent, then replacing  $\varphi$  with  $\psi$  will have no effect on the truth-value of a larger sentence.

*Formal Statement:*

Suppose that  $\varphi$  and  $\psi$  are logically equivalent, that  $\varphi$  is a subformula of  $\chi$ , and that  $\chi'$  is the formula just like  $\chi$ , except that all instances of  $\varphi$  are replaced with  $\psi$ . It follows that  $\chi$  and  $\chi'$  are logically equivalent.

(36) **Key Result: Dropping Operators (Quantifiers) from Our System**

- Informally speaking, what (35) says is that if  $\varphi$  and  $\psi$  are logically equivalent, then anything we can ‘express’ in FOL with  $\varphi$ , we can also ‘express’ with  $\psi$
- Now, note the logical equivalence we just proved in (34):  
 $\exists xPx$  is logically equivalent to  $\sim\forall x\sim Px$
- From (34), it follows that anything we can ‘express’ in FOL with ‘ $\exists$ ’ we can express with ‘ $\forall$ ’ and ‘ $\sim$ ’.
- **Consequently, we could drop ‘ $\exists$ ’ from our FOL system, leaving just ‘ $\forall$ ’ and ‘ $\sim$ ’, and still have an equally ‘expressive’ system**
- **In addition, we could simply view ‘ $\exists$ ’ as a *special abbreviation* for more complex formulae:**  
 $\exists v\psi$  is ‘shorthand’ for  $\sim\forall x\sim\psi$
- (We’ll make use of these ideas later, where they will come in handy for the definitions of logical languages we use for the semantic analysis of English...)

(37) **Entailment**

Let  $\varphi, \psi$  be sentences of FOL.  $\varphi$  entails  $\psi$  if every model  $\mathcal{M}$  such that  $[[\varphi]]^{\mathcal{M}} = 1$  is also such that  $[[\psi]]^{\mathcal{M}} = 1$

Illustration:  $(\forall x Px \vee \forall x Qx)$  entails  $\forall x(Px \vee Qx)$

Proof of Entailment:

Let  $\mathcal{M}$  be any model  $\langle D, I \rangle$  such that  $[(\forall x Px \vee \forall x Qx)]^{\mathcal{M}} = 1$ . Let  $g$  be any variable assignment based on  $\mathcal{M}$ .

- Thus,  $[(\forall x Px \vee \forall x Qx)]^{M,g} = 1$
- Thus, either (i)  $[(\forall x Px)]^{M,g} = 1$  or (ii)  $[(\forall x Qx)]^{M,g} = 1$
- Suppose (i). Then for all  $a \in D$ ,  $[(Px)]^{M,g(x/a)} = 1$ 
  - Therefore, for all  $a \in D$ , either  $[(Px)]^{M,g(x/a)} = 1$  or  $[(Qx)]^{M,g(x/a)} = 1$
  - Therefore, for all  $a \in D$ , either  $[(Px \vee Qx)]^{M,g(x/a)} = 1$
  - Therefore,  $[(\forall x(Px \vee Qx))]^{\mathcal{M}} = 1$
- Suppose (ii). Then for all  $a \in D$ ,  $[(Qx)]^{M,g(x/a)} = 1$ 
  - Therefore, for all  $a \in D$ , either  $[(Px)]^{M,g(x/a)} = 1$  or  $[(Qx)]^{M,g(x/a)} = 1$
  - Therefore, for all  $a \in D$ , either  $[(Px \vee Qx)]^{M,g(x/a)} = 1$
  - Therefore,  $[(\forall x(Px \vee Qx))]^{\mathcal{M}} = 1$
- Thus,  $[(\forall x(Px \vee Qx))]^{M,g} = 1$

*Now that we have this notion of ‘interpretation with respect to a model’, it becomes possible to prove rigorously that our proof system for FOL is sound and complete!*

(38) **Soundness of PL**

- If  $S \vdash \psi$ , then  $S \vDash \psi$
- If  $S \vdash \psi$ , then if  $\mathcal{M}$  is a model for  $S$ , then  $[[\psi]]^{\mathcal{M}} = 1$

(39) **Completeness of PL**

- If  $S \vDash \psi$ , then  $S \vdash \psi$
- If every model  $\mathcal{M}$  for  $S$  is also a model for  $\psi$ , then  $S \vdash \psi$

**Problem Set on Propositional Logic and First Order Logic:  
Formal Semantics<sup>1</sup>**

**(1) Computing Entailment in Propositional Logic**

Please use truth-tables to establish whether the following claims are accurate.

- a.  $\{ (\neg p \vee \neg q) \} \vDash \neg(p \vee q)$
- b.  $\{ (p \vee q), (\neg p \rightarrow \neg q) \} \vDash q$
- c.  $\{ ((p \rightarrow q) \rightarrow r) \} \vDash (p \rightarrow (q \rightarrow r))$

**(2) Computing Logical Equivalence in Propositional Logic**

Please use truth-tables to show that the following pairs are logically equivalent.

- |    |                                     |   |
|----|-------------------------------------|---|
| a. | (i) $(\varphi \& \psi)$             | (ii) $\neg(\neg\varphi \vee \neg\psi)$            |
| b. | (i) $(\varphi \rightarrow \psi)$    | (ii) $(\neg\psi \rightarrow \neg\varphi)$         |
| c. | (i) $(\varphi \& (\psi \vee \chi))$ | (ii) $((\varphi \& \psi) \vee (\varphi \& \chi))$ |

**(3) Defining Operators in Propositional Logic**

Please show how the operators ‘~’ and ‘v’ can be defined using the ‘nor’-operator ( $\downarrow$ ) defined below.

$\varphi$	$\psi$	$\ $	$(\varphi \downarrow \psi)$
1	1	$\ $	0
1	0	$\ $	0
0	1	$\ $	0
0	0	$\ $	1

**(4) Computing Truth of Formulas Relative to a Model**

Let  $\mathcal{M}$  be the model  $\langle L, I \rangle$ , where  $L$  is the set of English letters { a, b, c, d, e, f, g, h, I, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z } and  $I$  consists of at least the following mappings:

$$\begin{aligned} I(A) &= \{ x : x \in L \text{ and } x \text{ is a vowel} \} \\ I(R) &= \{ \langle x, y \rangle : x, y \in L \text{ and } x \text{ precedes } y \text{ in alphabetical order} \} \end{aligned}$$

Please calculate the truth-values of the following sentences relative to  $\mathcal{M}$ :

- a.  $\exists x \exists y \exists z (Rxy \& (Ay \& (Rxz \& \neg Az)))$
- b.  $\forall x (Rxx \rightarrow \neg Ax)$
- c.  $\forall x (\neg Ax \rightarrow \exists y Rxy)$

---

<sup>1</sup> Most of these exercises are taken from Gamut (1991), Volume 1: Chapter 2, Chapter 3

(5) **Proving Logical Equivalence for First Order Logic Sentences**

Please show that the following pairs of formulae are logically equivalent.

- |    |     |                           |      |                           |
|----|-----|---------------------------|------|---------------------------|
| a. | (i) | $\forall x \forall y Pxy$ | (ii) | $\forall y \forall x Pxy$ |
| b. | (i) | $\exists x \exists y Pxy$ | (ii) | $\exists y \exists x Pxy$ |

(6) **Proving Entailment for First Order Logic Sentences**

Please show that the following entailment relations hold.

- a.  $\forall x Px \vDash \exists x Px$ <sup>2</sup>
- b.  $\exists x(Px \ \& \ Qx) \vDash (\exists xPx \ \& \ \exists xQx)$
- c.  $\exists y \forall x Pxy \vDash \forall x \exists y Pxy$

---

<sup>2</sup> Hint: Recall that the domain D of a model has to be a *non-empty* set.

## Proving the Soundness and Completeness of Propositional Logic: Some Highlights<sup>1</sup>

### (1) A Summary of What We've Done So Far for PL

- a. We've given a purely *syntactic* characterization of 'valid inference' in PL  
 $S \vdash \psi$
- b. We've given a formal semantics for PL notation, and used it to provide a (proper) semantic definition of 'valid inference' in PL.  
 $S \vDash \psi$

### (2) The BIG Question:

Do these two characterizations of validity coincide?

$$S \vdash \psi \qquad \text{iff} (?) \qquad S \vDash \psi$$

### (3) The Theorems We Wish to Prove

#### Soundness of PL:

If  $\psi$  can be derived from  $S$  in our natural deduction system for PL, then  $S$  entails  $\psi$

- If  $S \vdash \psi$ , then  $S \vDash \psi$
- If  $S \vdash \psi$ , then if  $V$  is a valuation for  $S$ ,  $V(\psi) = T$

#### Completeness of PL:

If  $S$  entails  $\psi$ , then  $\psi$  can be derived from  $S$  in our natural deduction system.

- If  $S \vDash \psi$ , then  $S \vdash \psi$
- If every valuation  $V$  of  $S$  is also a valuation of  $\psi$ , then  $S \vdash \psi$

### (4) Some History of the Proofs

- Soundness was basically proven rather early on (it's easy, but tedious)
- Before there were proper proofs, people were largely convinced that PL and FOL were 'complete'
  - After all, anything anyone ever wanted to prove could be proved!
- The first proper proof that FOL is complete was Gödel's PhD thesis (1929)
  - It's crazy complicated, and nobody teaches it anymore
- In his PhD thesis, Henkin (1949) hit upon a much simpler, and just plain cooler proof
  - This is the one everybody teaches to this day...

---

<sup>1</sup> These notes are based upon material in the following required readings: Gamut (1991), Chapter 4 pp. 148-155; Crossley *et al.* (1972), Chapter 2; Partee *et al.* (1993) Chapter 8 pp. 225-227.

*For reasons of time, I won't give the complete proof of either soundness or completeness:*

- However, I will hit the 'highlights' of both...
- As we'll see, the completeness proof is more involved, *and more interesting!*...

## 1. Proving the Soundness of Natural Deduction for Propositional Logic

(5) **Theorem to Prove: Soundness**      If  $S \vdash \psi$ , then  $S \vDash \psi$

### (6) Key Observation

If  $S \vdash \psi$ , then there is a finite subset  $S' \subseteq S$  such that there is a derivation consisting of  **$n$  lines** where each  $\varphi \in S'$  appear as 'Assumptions' and where  $\psi$  appears on **line  $n$** .

- *Key Idea:*  
We can use (strong) induction to prove the following, which would prove (5):

### (7) Restatement of Soundness (for Mathematical Induction)

For every natural number  $n > 0$ , if  $S \vdash \psi$  with a proof consisting of  $n$  lines, then  $S \vDash \psi$

*I won't give the entire inductive proof of (7), but I'll give you the main gist...*

### (8) Preliminary Observation

The following are, technically speaking, proofs in our natural deduction system.

- a. 1. p                  Assumption
- b. 1. p                  Assumption  
2. (q & r)                  Assumption  
3. (s → t)                  Assumption

- Proof (1a) consists of one single line. It terminates right after we add 'p' as an assumption. Thus,  $\{p\} \vdash p$  (which intuitively should be the case)
- Proof (1b) has three lines. It terminates right after we add the third assumption ' $(s \rightarrow t)$ '. Thus,  $\{p, (q \& r), (s \rightarrow t)\} \vdash (s \rightarrow t)$  (which intuitively should be the case)

## (9) Proof of Soundness Theorem (7) By Strong Induction

### a. Base Step: $n = 1$

Suppose that  $S \vdash \psi$  with a proof consisting of 1 line.

- Thus, for some finite subset  $S' \subseteq S$ ,  $S' \vdash \psi$  with a proof consisting of 1 line.
- Given our system, the proof in question must be a ‘degenerate’ case like (8a), where  $\psi$  is an Assumption.
- Consequently,  $\psi \in S'$ , and so  $\psi \in S$ . Consequently,  $S \vDash \psi$

### b. Induction Step:

Let  $n \in \mathbb{N}$  be such that for all  $m < n$ , if  $S \vdash \psi$  with a proof consisting of  $m$  lines, then  $S \vDash \psi$ .

- We'll now show that if  $S \vdash \psi$  with a proof consisting of  $n$  lines, then  $S \vDash \psi$ .
- We'll show this by considering all the ways that a proof consisting of  $(n-1)$  lines can be extended to a proof consisting of  $n$  lines.
- Given the structure of our system, there are 12 cases to consider:
  1. Adding an assumption  $\psi$
  2. Deriving  $\psi$  by Repetition
  3. Deriving  $\psi$  by I&
  4. Deriving  $\psi$  by E&
  5. Deriving  $\psi$  by Iv
  6. Deriving  $\psi$  by Ev
  7. Deriving  $\psi$  by E $\rightarrow$
  8. Deriving  $\psi$  by I $\rightarrow$
  9. Deriving  $\psi$  by E~
  10. Deriving  $\psi$  by I~
  11. Deriving  $\psi$  by ~
  12. Deriving  $\psi$  by EFSQ

*For reasons of time, I won't do all 12...  
Just a few notable ones...*

#### 1. *Adding $\psi$ as an Assumption*

Suppose that  $S \vdash \psi$  with a proof consisting of  $n$  lines, where the final line has ‘Assumption’ as the justification. It follows that  $\psi \in S$ , and so  $S \vDash \psi$

2. *Deriving  $\psi$  by Repetition*

Suppose that  $S \vdash \psi$  with a proof consisting of  $n$  lines, where the final line has ‘Repetition’ as the justification.

- By definition of ‘Repetition’,  $S \vdash \psi$  with a proof of length  $m < n$ .
- Therefore, by the Induction Assumption,  $S \vDash \psi$ .

3. *Deriving  $\psi$  by I&*

Suppose that  $S \vdash \psi$  with a proof consisting of  $n$  lines, where the final line has ‘I&’ as the justification.

- By definition of ‘I&’,  $\psi = (\varphi \ \& \ \chi)$ , and  $S \vdash \varphi$  with a proof of length  $m < n$ , and  $S \vdash \chi$  with a proof of length  $m < n$
- Therefore, by the induction assumption,  $S \vDash \varphi$  and  $S \vDash \chi$
- Therefore,  $S \vDash (\varphi \ \& \ \chi) (= \psi)$

4. *Deriving  $\psi$  by E&*

(Can be shown via an argument parallel to the one for I&)

5. *Deriving  $\psi$  by Iv*

(easily shown via an argument similar to those above)

6. *Deriving  $\psi$  by Ev*

(easily shown via an argument similar to those above)

7. *Deriving  $\psi$  by E $\rightarrow$*

Suppose that  $S \vdash \psi$  with a proof consisting of  $n$  lines, where the final line has ‘E $\rightarrow$ ’ as the justification.

- By definition of ‘E $\rightarrow$ ’, it follows that  $S \vdash (\varphi \rightarrow \psi)$  with a proof of length  $m < n$ , and  $S \vdash \varphi$  with a proof of length  $m' < n$ .
- Therefore, by the induction assumption,  $S \vDash (\varphi \rightarrow \psi)$  and  $S \vDash \varphi$ .
- Therefore  $S \vDash \psi$ .<sup>2</sup>

*The other steps in the proof are basically parallel to this...*

*EXCEPT THAT: the steps for I $\rightarrow$  and I~ rely upon a minor (trivial) lemma concerning ‘conditional proofs’...*

<sup>2</sup> After all, if there were a valuation V of S s.t.  $V(\psi) = 0$ , then since  $S \vDash \varphi$ , this valuation would be s.t.  $V(\varphi \rightarrow \psi) = 0$ , and so S wouldn’t entail  $(\varphi \rightarrow \psi)$ .

(10) **Important Note**

As trivial as it is, this proof of the soundness of PL would not even get off the ground without a clear, mathematically precise definition of what ‘entailment’ for PL is...

- And this requires a clear, mathematically precise definition of what an ‘interpretation’ of PL is...

---

## 2. Proving the Completeness of Natural Deduction for Propositional Logic

(11) **Theorem to Prove: Completeness**      If  $S \vDash \psi$ , then  $S \vdash \psi$ .

*The first crucial step to proving completeness is the ‘Key Lemma’ in (13).*

- For reasons of time, I won’t review the demonstration here.
- Interested readers are referred to Gamut (1991), p. 150

(12) **Key Preliminary Definition: Consistency**

Let  $S$  be a set of formulae in PL.  $S$  is *inconsistent* if  $S \vdash \perp$ .  $S$  is *consistent* if  $S \not\vdash \perp$

- Note that ‘(in)consistency’ here is a *syntactic* property.

(13) **Key Lemma**

Let  $S$  be a set of formulae in PL.  $S \cup \{\psi\}$  is inconsistent iff  $S \vdash \sim\psi$

- Note that (13) just states that  $S \cup \{\psi\} \vdash \perp$  iff  $S \vdash \sim\psi$
- If you consider our rules of  $I\sim$  and  $E\sim$ , you can see that (13) pretty trivially holds...

*The second crucial step to proving completeness is seeing how the ‘Consistency Theorem’ in (14) would entail Completeness in (11)*

(14) **The Consistency Theorem**

If  $S$  is a consistent set of formulae in PL, then there is a valuation  $V$  of  $S$ .

(15) **The Consistency Theorem Entails Completeness**

- Suppose that  $S \vDash \psi$ . It follows that  $S \cup \{\sim\psi\}$  has **no valuation**.
- Therefore, by contraposition of (14), it follows that  $S \cup \{\sim\psi\}$  is **inconsistent**.
- Therefore, by (13), it follows that  $S \vdash \sim\psi$ , and so  $S \vdash \psi$ .

The third and most arduous step in the completeness is proof is proving (14).  
And, the most arduous part of proving (14) is proving the lemma in (16)...

(16) **Lindenbaum's Lemma**

Let  $S$  be a consistent set of formulae in PL. There is a consistent set  $S^*$  such that  $S \subseteq S^*$  and  $S^*$  has the following key ‘closure properties’.<sup>3</sup>

For any formulae  $\varphi$  and  $\psi$  of PL:

- a.  $\varphi \in S^*$  iff  $\neg\varphi \notin S^*$
- b.  $(\varphi \& \psi) \in S^*$  iff  $\varphi \in S^*$  and  $\psi \in S^*$
- c.  $(\varphi \vee \psi) \in S^*$  iff  $\varphi \in S^*$  or  $\psi \in S^*$
- d.  $(\varphi \rightarrow \psi) \in S^*$  iff  $\varphi \notin S^*$  or  $\psi \in S^*$

Note:

For those who are interested, proving (16) isn’t intellectually all that difficult. It just takes time to correctly lay out the procedure for constructing  $S^*$  from  $S$ ....

*Now that we have this huge set  $S^*$ , we’re home free!*

(17) **The Cool Central Insight of Henkin’s Proof**

You can take a set  $S^*$  with the properties in (16), and *directly build a valuation for  $S^*$  from the formulas in  $S^*$  itself!*

(18) **Model Existence Lemma**

If a consistent set  $S^*$  has the ‘closure properties’ in (16a-d), then  $S^*$  has a valuation. Namely, it has the valuation defined as follows:

a. The Valuation for  $S^*$ :

Let the valuation  $V$  be such that for every proposition letter  $\alpha$  of PL,  $V(\alpha) = 1$  iff  $\alpha \in S^*$

b. Claim:

The valuation  $V$  defined in (18a) is a valuation for  $S^*$ .

*The final step in the completeness theorem is proving the claim in (18).*

- The proof will be by induction on the complexity of formulae...

*Again, I won’t do the whole proof here, but I’ll review some key illustrative steps...*

---

<sup>3</sup> This set  $S^*$  is commonly referred to as a ‘maximally consistent set’. Note, too, that since  $S^*$  is consistent,  $\perp \notin S^*$ .

(19) **Proof of the Model Existence Lemma**

Claim:

Let  $V$  be the valuation defined in (18a). For any formula  $\varphi$  of PL,  $\varphi \in S^*$  iff  $V(\varphi) = 1$ .

Proof (by Induction on Complexity of Formulae):

a. *Base Step: Proposition Letters*

Suppose that  $\varphi$  is a proposition letter. Then, by the very definition in (18a), it trivially follows that  $V(\varphi) = 1$  iff  $\varphi \in S^*$

b. *Induction Step:*

Suppose that  $\varphi$  is a (complex) formula of PL, and that for any of its immediate subformulae  $\psi \in S^*$  iff  $V(\psi) = 1$ . We will now show that  $\varphi \in S^*$  iff  $V(\varphi) = 1$ . There are four cases to consider:  $\sim, \&, \vee, \rightarrow$

1.  $\varphi = \sim\psi$

- $\sim\psi \in S^*$       iff      (by closure property in (16a))
- $\psi \notin S^*$       iff      (by induction assumption)
- $V(\psi) = 0$       iff      (by definition of a valuation)
- $V(\sim\psi) = 1$

2.  $\varphi = (\psi \& \chi)$

- $(\psi \& \chi) \in S^*$       iff      (by closure property in (16b))
- $\psi \in S^*$  and  $\chi \in S^*$       iff      (by induction assumption)
- $V(\psi) = 1$  and  $V(\chi) = 1$       iff      (by definition of a valuation)
- $V(\psi \& \chi) = 1$

3.  $\varphi = (\psi \vee \chi)$

*Proof is parallel to those for 1. and 2.*

4.  $\varphi = (\psi \rightarrow \chi)$

*Proof is parallel to those for 1. and 2.*

(20) **Putting It All Together**

- a. Given the lemma in (16), we've shown that any consistent set  $S$  can be 'expanded' into a larger consistent set  $S^*$  with the properties in (16a-d).
- b. Given (18)-(19), we've shown that any such set  $S^*$  with the properties in (16a-d) has a valuation  $V$ .
- c. Since  $V$  is a valuation for  $S^*$ , and  $S \subseteq S^*$ , it follows that  $V$  is a valuation for  $S$ .
- d. **Thus, any consistent set of formulae has a valuation  $V$ . QED (14).**

(21) **Taking Stock of What We've Done**

- In our last set of notes, we developed a mathematically rigorous characterization of what it means for a formula of PL to be ‘true under an interpretation’ (valuation).
- We’ve just seen how these notions have allowed us to prove that our syntactic proof system for PL *is a perfect syntactic characterization of validity in PL*
- **For the first time in human history, we’ve shown that we can indeed give a perfect, purely syntactic characterization of what it means for an inference to be valid (in a specified language)**
  - That’s a huge achievement...
  - And it’s even more an achievement when we do it for FOL...

**Key Applications of Model Theoretic Semantics for FOL:**  
**Soundness, Completeness, Compactness, etc.**<sup>1</sup>

**(1) A Summary of What We've Done So Far for PL**

- a. We've given a purely *syntactic* characterization of 'valid inference' in FOL  
 $S \vdash \psi$
- b. We've given a formal (model-theoretic) semantics for FOL notation, and used it to provide a (proper) semantic definition of 'valid inference' in FOL.  
 $S \vDash \psi$

*In these notes, I'll review some key applications of our model-theoretic semantics for FOL.*

*These results will provide further motivation for using 'models' as a mathematical characterization of 'interpretation' for FOL...*

---

**1. Proving Soundness and Completeness for First Order Logic**

**(2) The Theorems We Wish to Prove**

Soundness of FOL:

If  $\psi$  can be derived from  $S$  in our natural deduction system for FOL, then  $S$  entails  $\psi$

- If  $S \vdash \psi$ , then  $S \vDash \psi$
- If  $S \vdash \psi$ , then if  $\mathcal{M}$  is a model for  $S$ ,  $[[\psi]]^{\mathcal{M}} = T$

Completeness of FOL:

If  $S$  entails  $\psi$ , then  $\psi$  can be derived from  $S$  in our natural deduction system.

- If  $S \vDash \psi$ , then  $S \vdash \psi$
- If every model  $\mathcal{M}$  for  $S$  is also a model for  $\psi$ , then  $S \vdash \psi$

**(3) On the Proof of Soundness for FOL**

The proof of soundness for FOL is not importantly different from the proof of soundness for PL.

- We simply extend the soundness proof of PL so that the induction step also considers the cases where:

- |       |                                 |      |                                 |
|-------|---------------------------------|------|---------------------------------|
| (i)   | $\psi$ is derived by $E\exists$ | (ii) | $\psi$ is derived by $I\exists$ |
| (iii) | $\psi$ is derived by $E\forall$ | (iv) | $\psi$ is derived by $I\forall$ |

---

<sup>1</sup> These notes are based upon material in the following required readings: Gamut (1991), Chapter 4 pp. 148-155; Crossley *et al.* (1972), Chapter 2; Partee *et al.* (1993) Chapter 8 pp. 198-201.

(4) **Illustration of the Additional Steps for Proving Soundness of FOL**

The following subcase can be added to the induction step in (9b) of the handout “Proving the Soundness and Completeness of Propositional Logic”

13. *Deriving  $\psi$  by  $I\exists$*

Suppose that  $S \vdash \psi$  with a proof consisting of  $n$  lines, where the final line has ‘ $I\exists$ ’ as the justification.

- By definition of ‘ $I\exists$ ’,  $\psi = \exists x\varphi$  and  $S \vdash [\alpha/x]\varphi$  with a proof of length  $m < n$ .
- By the induction assumption, then  $S \vDash [\alpha/x]\varphi$ .
- Now, let  $\mathcal{M}$  be any model  $\langle D, I \rangle$  for  $S$ , and  $g$  be any variable assignment based on  $\mathcal{M}$ .
- It follows that  $\mathcal{M}$  is a model for  $[\alpha/x]\varphi$ .
- Therefore  $[[ [\alpha/x]\varphi ]]^{M,g} = 1$
- **Let  $I(\alpha) = a$ . It follows that  $[[ \varphi ]]^{M,g(x/a)} = 1$ .**
- Therefore, there is an  $a \in D$  such that  $[[ \varphi ]]^{M,g(x/a)} = 1$
- Therefore,  $[[ \exists x\varphi ]]^{M,g} = 1$ .
- Since  $g$  was arbitrary, it follows that  $[[ \exists x\varphi ]]^M = 1$ , and so  $\mathcal{M}$  is also a model for  $\exists x\varphi$ .
- Since  $\mathcal{M}$  was arbitrary, it follows that any model of  $S$  is also a model of  $\exists x\varphi$ , and so  $S \vDash \exists x\varphi$ ,

*For reasons of time, I won't walk through the other four additional cases...*

**Nevertheless, we can still see that this proof wouldn't even get off the ground without our formal, model-theoretic semantics for FOL...**

(4) **On the Proof of Completeness for FOL**

The proof of completeness for FOL has the same general structure as the proof of completeness for PL.

a. Key Lemma:

Let  $S$  be a set of sentences in FOL.  $S \cup \{\psi\}$  is inconsistent iff  $S \vdash \sim\psi$

b. Consistency Theorem:

If  $S$  is a consistent set of sentences in FOL, then there is a model  $\mathcal{M}$  of  $S$ .

c. Consistency Theorem Entails Completeness:

- If  $S \vDash \psi$ , then  $S \cup \{\sim\psi\}$  has **no model**.
- Therefore, consistency theorem (4b) entails that  $S \cup \{\sim\psi\}$  is inconsistent.
- Therefore, key lemma (4a) entails that  $S \vdash \sim\psi$ , and so  $S \vdash \psi$

Again, the most arduous step of the completeness proof is showing proving the Consistency Theorem (4b)...

And the most difficult part of proving (4b) is proving the lemma in (5)...  
**Also, proving this lemma is much more involved for FOL than for PL.**

(5) **Lindenbaum's Lemma**

Let  $S$  be a consistent set of sentences in FOL. There is a consistent set  $S^*$  such that  $S \subseteq S^*$  and  $S^*$  has the following key ‘closure properties’.<sup>2</sup>

For any formulae  $\varphi$  and  $\psi$  of PL:

- a.  $\varphi \in S^*$  iff  $\neg\varphi \notin S^*$
- b.  $(\varphi \& \psi) \in S^*$  iff  $\varphi \in S^*$  and  $\psi \in S^*$
- c.  $(\varphi \vee \psi) \in S^*$  iff  $\varphi \in S^*$  or  $\psi \in S^*$
- d.  $(\varphi \rightarrow \psi) \in S^*$  iff  $\varphi \notin S^*$  or  $\psi \in S^*$
- e.  $\exists x\varphi \in S^*$  iff there is an individual constant  $b$  such that  $[b/x]\varphi \in S^*$
- f.  $\forall x\varphi \in S^*$  iff for every individual constant  $b$ ,  $[b/x]\varphi \in S^*$

Again, now that we have this huge set  $S^*$ , we're home free!...

(6) **The Cool Central Insight of Henkin's Proof**

You can take a set  $S^*$  with the properties in (5), and *directly build a model for  $S^*$  from the formulas in  $S^*$  itself!*

(7) **The Model Existence Lemma**

If a consistent set  $S^*$  of FOL sentences has the closure properties in (5), then there is a model  $\mathcal{M}$  for  $S^*$ .

a. The Model for  $S^*$

Let  $\mathcal{M}$  be the model  $\langle D, I \rangle$  where:

(i)  $D$  is the set of all the individual constants in the language.

(ii) The function  $I$  satisfies the conditions below.

1. If  $\alpha$  is an individual constant, then  $I(\alpha) = \alpha$

2. If  $\Phi$  is a predicate letter of arity  $n$ , then  $I(\Phi)$  is such that:

$$\langle \alpha_1, \dots, \alpha_n \rangle \in I(\Phi) \text{ iff } \Phi\alpha_1 \dots \alpha_n \in S^*$$

b. Key Claim: The model  $\mathcal{M}$  defined in (7a) is a model for  $S^*$

<sup>2</sup> Those who are intimately familiar with the completeness proof for FOL will know that I'm ‘fudging’ here on the statement of Lindenbaum's Lemma for FOL (since I'm not mentioning extending our FOL language by adding infinitely many constants).

To prove the key claim in (7), and thus the Model Existence Lemma, we again do (strong) induction on the number of logical operators in an FOL sentence...

- The proof is basically the same as that for PL; the only new and interesting steps come with the atomic formulae and the quantificational formulae

## (8) Proof of the Model Existence Lemma

### Claim:

Let  $\mathcal{M}$  be the model defined in (7a). For any natural number  $n$ , if  $\varphi$  is a sentence of FOL with  $n$  logical operators, then  $\varphi \in S^*$  iff  $[[\varphi]]^M = 1$ .

### Proof (by Induction on Complexity of Formulae):

#### a. Base Step: Atomic Sentences ( $n = 0$ )

Suppose that  $\varphi$  is an atomic formula  $\Phi\alpha_1 \dots \alpha_n$ .

- $\Phi\alpha_1 \dots \alpha_n \in S^*$       iff      (by condition 2 in definition of  $\mathcal{M}$ )
- $\langle \alpha_1, \dots, \alpha_n \rangle \in I(\Phi)$       iff      (by condition 1 in definition of  $\mathcal{M}$ )
- $\langle I(\alpha_1), \dots, I(\alpha_n) \rangle \in I(\Phi)$       iff      (by definition of a ‘model’)
- $[[\Phi\alpha_1 \dots \alpha_n]]^M = 1$

#### b. Induction Step:

Let  $n$  be such that for all  $m < n$ , if  $\varphi$  is a sentence of FOL with  $m$  logical operators, then  $\varphi \in S^*$  iff  $[[\varphi]]^M = 1$ . We will now show that if  $\varphi$  is a sentence of FOL with  $n$  logical operators, then  $\varphi \in S^*$  iff  $[[\varphi]]^M = 1$ .

There are six cases to consider:  $\sim, \&, \vee, \rightarrow, \exists, \forall$

1.  $\varphi = \sim\psi$       Same as in completeness proof for PL

2.  $\varphi = (\psi \& \chi)$       Same as in completeness proof for PL

3.  $\varphi = (\psi \vee \chi)$       Same as in completeness proof for PL

4.  $\varphi = (\psi \rightarrow \chi)$       Same as in completeness proof for PL

5.  $\varphi = \exists x\psi$

- $\exists x\psi \in S^*$       iff      (by closure property (5e))

- There is an individual constant  $b$  such that  $[b/x]\psi \in S^*$

iff (by the induction assumption)

- $[[ [b/x]\psi ]]^M = 1$       iff

- $[[ \psi ]]^{M,g(x/b)} = 1$ , for an arbitrary variable assignment  $g$       iff

- $[[\exists x\psi]]^M = 1$

6.  $\varphi = \forall x\psi$       *Proof similar to the one in 5.*

(9) **Putting it All Together**

- a. Given the lemma in (5), any consistent set  $S$  of FOL sentences can be ‘expanded’ into a larger consistent set  $S^*$  with the properties in (5a-f).
- b. Given (7)-(8), any such set  $S^*$  with the properties in (5a-f) has a model  $\mathcal{M}$ .
- c. Since  $\mathcal{M}$  is a model for  $S^*$ , and  $S \subseteq S^*$ , it follows that  $\mathcal{M}$  is a model for  $S$ .  
**Thus, any consistent set of FOL sentences has a model  $\mathcal{M}$ . QED (4b).**

(10) **Really Cool Thing to Notice**

In the Henkin proof of FOL’s completeness, we *construct* a model structure directly from the set of FOL sentences  $S^*$ .

- The domain is the set of individual constants
- Each constant is interpreted as itself
- Each predicate letter  $\Phi$  is interpreted as the relation holding of  $\alpha_1, \dots, \alpha_n$  iff the sentence ‘ $\Phi\alpha_1 \dots \alpha_n$ ’ is in  $S^*$

Thus, the very nature of models themselves – objects of the form  $\langle D, I \rangle$  - factor into the central core step of the proof.

- This could be viewed as giving ‘additional motivation’ for this particular formalization of the notion of ‘FOL interpretation.’

(11) **Taking Stock of What We’ve Done**

- In our last set of notes, we developed a mathematically rigorous characterization of what it means for a sentence of FOL to be ‘true under an interpretation’ (model).
- We’ve just seen how this notion has allowed us to prove that our syntactic proof system for FOL is a perfect syntactic characterization of validity in FOL
- **For the first time in human history, we’ve shown that we can indeed give a perfect, purely syntactic characterization of what it means for an inference to be valid (in a specified language)**

## 2. Some Other Important Results of our Model Theoretic Semantics

### (12) Demonstrating Consistency

Suppose you want to know whether some set  $S$  of FOL formulae are *consistent* or not; that is, you want to know whether  $S \vdash \perp$  (inconsistent) or  $S \not\vdash \perp$  (consistent).

#### a. The Challenge:

If all we have is the natural deduction system, if  $S \vdash \perp$ , then we can eventually show that (we'll have the proof). But if  $S \not\vdash \perp$ , there's no way to conclusively show this (in finite time) with just the natural deduction system.

#### b. The Solution:

- Given soundness, if  $S \vdash \perp$ , then  $S \models \perp$ , and so there is no model  $\mathcal{M}$  for  $S$ .
- Thus, with our model theoretic semantics, we can show that  $S$  is consistent ( $S \not\vdash \perp$ ) by devising a model for  $S$ !**

### (13) Demonstrating Independence

Suppose you want to know whether some FOL sentence  $\varphi$  can be derived from  $S$  or not. That is, you want to know whether  $S \vdash \varphi$  or  $S \not\vdash \varphi$ .

- In the latter case, we say that  $\varphi$  is 'independent' of  $S$ .

#### a. The Challenge:

Again, if all we have is the natural deduction system, there's no way to conclusively show (in finite time) that  $S \not\vdash \varphi$ .

#### b. The Solution:

Given soundness again, if  $S \not\vdash \varphi$  then  $S \not\models \varphi$ . Therefore, if we can show that there is a model  $\mathcal{M}$  of  $S$  such that  $[[\varphi]]^{\mathcal{M}} = 0$ , we've shown that  $S \not\vdash \varphi$ .

### (14) Historical Relevance of (12) and (13)

For centuries, mathematicians struggled to show whether the fifth axiom of Euclid's geometry was 'independent' of the other axioms (or not).

- People tried to derive contradictions from the negation of the fifth axiom, but couldn't succeed.
- Finally, however, mathematicians succeeded in showing that there were (informal) models of geometric systems where (only) the fifth axiom is negated.
  - Such (informal) models show that non-Euclidean geometries are consistent (12) and that the fifth axiom is indeed independent (13).

#### (14) Compactness Theorem

a. Claim:

A set of FOL sentences  $S$  has a model *iff* every finite subset of  $S$  has a model.

b. Proof:

(i) If  $S$  has a model, then of course every finite subset does (duh).

(ii) Suppose  $S$  doesn't have a model.

- Thus,  $S \models \perp$  and so by completeness,  $S \vdash \perp$ .
- Since derivations are finite, it follows that there is some finite subset  $S' \subseteq S$  such that  $S' \vdash \perp$ .
- Thus, by soundness,  $S' \models \perp$ , and so  $S'$  doesn't have a model.
- Thus, *not* every finite subset  $S' \subseteq S$  has a model.

*The theorem in (14) is a powerful tool in advanced meta-logic and model-theory, since it allows for easier proofs that certain infinite sets of sentences are consistent...*

#### (15) The Löwenheim-Skolem Theorems

a. The ‘Downward’ Löwenheim-Skolem Theorem

If there is a model  $\mathcal{M} (= \langle D, I \rangle)$  for  $S$ , then there is a model  $\mathcal{M}' (= \langle D', I' \rangle)$  for  $S$  such that  $D'$  is countable (finite or countably infinite).

b. The ‘Upward’ Löwenheim-Skolem Theorem

If there is a model  $\mathcal{M} (= \langle D, I \rangle)$  for  $S$  whose domain  $D$  is countably infinite, then there is a model  $\mathcal{M}' (= \langle D', I' \rangle)$  for  $S$  such that  $D'$  is *uncountable*.

*The theorems in (15) entail that – although sentences of FOL can ‘say’ that there are infinitely many things – they cannot say whether that infinity is countable or not...*

- Some philosophers (Putnam) have also tried to connect (15) with philosophical problems relating to the nature of reference and meaning...

#### (16) Lindstrom’s Theorem (Informally Put)

Any logic that satisfies conditions C (unnamed here), and satisfies compactness (14), and also satisfies the Downward Löwenheim-Skolem Theorem *just is* FOL.

*Lindstrom’s Theorem provides a powerful tool for showing that a given logical system – no matter how superficially different from FOL it is – is ultimately just a notational variant of FOL.*

### 3. Models Beyond First Order Logic

As shown above, the introduction of ‘models’ as abstract characterizations of ‘interpretation’ for FOL has been an extremely important and fertile development in logic and mathematics...

- Throughout the 50s and 60s, logicians developed model-theoretic semantics for logical systems beyond FOL.

#### (17) Modal Logic

One of the first huge advances was Saul Kripke’s development of a model theoretic semantics for modal logic, along with the first completeness proofs of such logics.

- Syntax: PL/FOL + the unary sentence connectives  $\diamond$  ‘it is possible that’ and  $\Box$  ‘it is necessary that’  
 $\diamond(p \ \& \ \Box q)$  ‘it’s possible that  $p$  and it’s necessary that  $q$ ’.
- Semantics: We add to the model structure a set of possible worlds (and a relation over those worlds, a.k.a. ‘the modal base’)

#### (18) Temporal Logic

- Syntax: PL/FOL + the unary sentence connectives P ‘it was the case that’ and F ‘it will be the case that’  
 $F(p \ \& \ Pq)$  ‘it will be the case that  $p$  and it was the case that  $q$ .
- Semantics: We add to the model structure a set of times (and a linear ordering over those times)

#### (19) Second Order Logic

- Syntax: FOL + variables over n-ary relations (for every natural number n)  
 $\forall P (Pab \rightarrow Pbc)$   
‘every relation that holds between a & b also holds between c & b’
- Semantics: We extend variable assignments so that they can accommodate these new variables.

*When we combine these notations and model-theoretic structures, we seem to be approaching the expressive capacity of a significant sub-part of human language!*

‘It was once possible that Dave would have something in common with Mary’  
 $P\diamond F\exists P (Pd \ \& \ Pm)$

(20) **The Exciting Possibility (ca. 1950s and 1960s)**

Could we develop a model theoretic semantics for a natural language (or at least a significant part of one)?

(21) **Why Would We Want a Model-Theoretic Semantics for English?**

a. A Compositional Semantics for Natural Language

A model theoretic semantics *is* a compositional semantics.

- The definition of ‘interpretation w.r.t. a model  $\mathcal{M}$  specifies how the semantic value of a complex expression is computed in terms of (i) the semantic value of its component expressions, and (ii) how the complex expression is syntactically constructed.
- Why would we want a compositional semantics for a natural language? See Linguistics 610....

b. A Theory of How Language Connects with ‘Reality’

Interpreting a language w.r.t. a model gives us a theory of how the language ‘hooks up’ with language-external reality.

- A model is a little mathematical representation of the universe (or ‘multiverse’, if we have possible worlds).
  - The domain D are the entities that exist.
  - The range of I gives us properties and relations between those entities
- The function I in the model maps linguistic expressions to ‘things’ in the model of reality:
  - Individual constants are mapped to entities
  - Predicates are mapped to properties and relations holding of them
- Why would we want a semantic theory that maps language to language/mind-external reality? (See Intro to Philosophy of Language)

c. Natural Languages are Not Fundamentally Different from Artificial Languages

- For most of the 20<sup>th</sup> century, it was commonly held that there is a fundamental difference between natural languages and artificial languages (like FOL).
- This difference in type was held to entail that the mathematical tools for analyzing artificial languages could not be applied to natural languages.
- **But, if we could give a model theoretic semantics for NL, that would weaken the idea that there is such a fundamental division (Montague)**

## (22) Another, Partisan Comment

- When we semanticists lay out a model theoretic semantics for a natural language, **we thereby specify what (some of) the valid inferences in that language are.**
  - Under any model theoretic semantics, some sentences will end up being predicted to be true whenever some other (set of) sentences are...
- Moreover, in as much as the theory is truly rigorous and formal, **these predictions can be calculated without any understanding of the (object) language itself.**

**Thus, we semanticists are the true inheritors of the original Aristotelian logical program: providing a formal/predictive characterization of valid inference for (natural) language.**

## (23) Immediate Challenges For Developing a Model-Theoretic Semantics for English

Alas, in the world of the late 50s and early 60s, the following were some obvious roadblocks to developing such a program for natural language semantics...

- a. Semantic Ambiguity of (Seemingly) Syntactically Unambiguous Sentences
  - Sentences like the following are semantically ambiguous, but they don't seem (at first glance) to be *syntactically* ambiguous.
    - (i) Some girl loves every man.
    - (ii) John wants to buy an ugly car.
  - A model-theoretic semantics, though, will map every sentence structure to exactly one interpretation in the model...
- b. The Ungodly Complexity of Natural Language Syntax
  - A model-theoretic semantics builds upon a recursive characterization of the sentences of a language (e.g. FOL).
  - Thus, to give a model-theoretic semantics for English, *we need some kind of recursive characterization of a 'well-formed' English sentence...*
  - *But who in the world understands English grammar well enough to give a recursive formal syntax for English?*

c. The Ungodly Complexity of the Syntax/Semantics Interface in NL

The main reason why people were so pessimistic about a formal semantics for natural language is that there just seemed to be so many *puzzles* about the syntax/semantics interface in natural language.

- (i) John wants to find a unicorn and eat it.
  - This sentence doesn't commit us to there being a unicorn...
  - So what the heck does 'it' refer to?
  - Could 'it' function as a bound pronoun? *How?*
- (ii) Every boy sings or dances.
  - This sentence doesn't mean the same thing as 'every boy sings or every boy dances'.
  - Thus, this isn't a case of two conjoined sentences with ellipsis.
  - So, 'or' must join together the two VPs directly...
  - *BUT HOW IS THAT POSSIBLE, IF IT ALSO JOINS TOGETHER TWO SENTENCES?*
- (iii) The temperature is 90 degrees and rising.
  - The predicate "is 90 degrees" seems to want the *extension* of "the temperature". (~ the extension of "the temperature" at this time is 90).
  - The predicate "is rising" seems to want the *intension* of "the temperature".
  - But, in this sentence, the NP "the temperature" is argument to *both* those predicates simultaneously... *HOW?!*
- (iv) A man who is smoking walked in.
  - The truth-conditions of this sentence seem to be:  
 $\exists x((Mx \ \& \ Sx) \ \& \ Wx)$   
'there is an x s.t. x is a man and x is smoking and x walked in.'
  - Clearly, 'Mx' is contributed by 'man' and 'Wx' by 'walked in...'
    - But how does 'who is smoking' contribute '(... & Sx)'?
    - *HOW DOES THAT WORK??*

**Of course, Montague (1974) showed how the issues in (23a-c) can be solved...  
To see how he did this, though, we need to start with ALGEBRA...**

## **Unit 3:**

### **Algebras and Semantics**

## An Algebraic Perspective on Propositional Logic: The First Steps Towards Montague Grammar<sup>1</sup>

- Many of the key ideas underlying Montague's general semantic program stem from an 'algebraic' perspective on logic and its formal semantics.
- In these notes, we will begin developing such an algebraic perspective.
- **However, it will require us to first make some superficial revisions to our syntax and semantics for PL and FOL.**

### 1. Prolegomena, Part 1: A New Presentation of PL

Recall our formal definition of a 'WFF' in Propositional Logic, below:

#### (1) The Definition of a 'Well-Formed Formula' (WFF) of PL

The set of 'well-formed formulae' of PL, WFF, is the smallest set such that:

- a. If  $\varphi$  is a proposition letter, then  $\varphi \in \text{WFF}$
- b. If  $\varphi, \psi \in \text{WFF}$ , then
  1.  $\sim\varphi \in \text{WFF}$
  2.  $(\varphi \ \& \ \psi) \in \text{WFF}$
  3.  $(\varphi \ \vee \ \psi) \in \text{WFF}$
  4.  $(\varphi \rightarrow \psi) \in \text{WFF}$

#### (2) A More Precise Restatement of the Conditions in (1b)

- a. If  $\varphi$  is a string in the vocabulary of PL which is in the set WFF, then the following is also in WFF: **The result of concatenating the symbol '¬' and the string  $\varphi$**
- b. If  $\varphi$  and  $\psi$  are strings in the vocabulary of PL which are in the set WFF, then the following is also in WFF: **The result of concatenating '(',  $\varphi$ , '&',  $\psi$  and ')'**.
- c. If  $\varphi$  and  $\psi$  are strings in the vocabulary of PL which are in the set WFF, then the following is also in WFF: **The result of concatenating '(',  $\varphi$ , '∨',  $\psi$  and ')'**.
- d. If  $\varphi$  and  $\psi$  are strings in the vocabulary of PL which are in the set WFF, then the following is also in WFF: **The result of concatenating '(',  $\varphi$ , '→',  $\psi$  and ')'**.

---

<sup>1</sup> These notes are based upon material in the following required readings: Partee *et al.* (1993) Chapter 9, Chapter 13 pp. 331-336. In addition, students are highly encouraged to begin reading Halvorsen & Ladusaw (1979) and Dowty *et al.* (1981) Chapter 8, and to start gradually working through Montague's original "Universal Grammar" paper.

Just for fun – but with big implications for later – let's introduce the ‘syntactic operations’ below:

(3) **Syntactic Operations Over the Vocabulary of PL**

The following operations (functions) take as input either strings of symbols over the vocabulary of PL or pairs of such strings:

- a. The Operation ‘Not’

$\text{Not}(\varphi) = \text{The result of concatenating the symbol ‘} \sim \text{’ and the string } \varphi$

- b. The Operation ‘And’

$\text{And}(\varphi, \psi) = \text{The result of concatenating ‘(’, } \varphi, \text{ ‘&’, } \psi \text{ and ‘)’}$

- c. The Operation ‘Or’

$\text{Or}(\varphi, \psi) = \text{The result of concatenating ‘(’, } \varphi, \text{ ‘v’, } \psi \text{ and ‘)’}$

- d. The Operation ‘If’

$\text{If}(\varphi, \psi) = \text{The result of concatenating ‘(’, } \varphi, \text{ ‘} \rightarrow \text{’, } \psi \text{ and ‘)’}$

With all this in place, we can offer the following equivalent restatement of the syntax of PL:

(4) **The Definition of a ‘Well-Formed Formula’ (WFF) of PL**

The set of ‘well-formed formulae’ of PL, WFF, is the smallest set such that:

- a. If  $\varphi$  is a proposition letter, then  $\varphi \in \text{WFF}$

- b. If  $\varphi \in \text{WFF}$ , then  $\text{Not}(\varphi) \in \text{WFF}$

- c. If  $\varphi, \psi \in \text{WFF}$ , then  $\text{And}(\varphi, \psi) \in \text{WFF}$

- d. If  $\varphi, \psi \in \text{WFF}$ , then  $\text{Or}(\varphi, \psi) \in \text{WFF}$

- e. If  $\varphi, \psi \in \text{WFF}$ , then  $\text{If}(\varphi, \psi) \in \text{WFF}$

(5) **Vocabulary**

Let  $A$  be a set, and  $f$  be an  $n$ -ary function. We say that  $A$  is *closed under*  $f$  if

- (i) Every  $\langle a_1, \dots, a_n \rangle \in A^n$  is in the domain of  $f$

- (ii) If  $\langle a_1, \dots, a_n \rangle \in A^n$ , then  $f(\langle a_1, \dots, a_n \rangle) \in A$

(6) **Observation**

The set WFF defined in (4) is closed under the operations Not, And, Or, and If.

(7) **Abstracting Even Further**

If we wanted to – and we will want to later on – we could schematically represent each of the ‘syntactic rules’ in (4b-e) as the following tuples:

- b.  $\langle \text{Not}, \langle \text{WFF} \rangle, \text{WFF} \rangle$   
‘The result of applying Not to a member of WFF is a WFF’
- c.  $\langle \text{And}, \langle \text{WFF}, \text{WFF} \rangle, \text{WFF} \rangle$   
‘The result of applying And to a pair consisting of a WFF and a WFF is a WFF’
- d.  $\langle \text{Or}, \langle \text{WFF}, \text{WFF} \rangle, \text{WFF} \rangle$   
‘The result of applying Or to a pair consisting of a WFF and a WFF is a WFF’
- e.  $\langle \text{If}, \langle \text{WFF}, \text{WFF} \rangle, \text{WFF} \rangle$   
‘The result of applying If to a pair consisting of a WFF and a WFF is a WFF’

(8) **Key Observation**

- In this way of describing the syntax of PL, we importantly distinguish between:
  - (i) Syntactic *Operations*
  - (ii) Syntactic *Rules*
- Syntactic Operations (Neg, And, Or, If) freely apply to any strings of symbols, and output strings that aren’t necessarily part of the language we want to define.
- Syntactic Rules (7b-e) make use of the syntactic operations to define **categories** of strings in the vocabulary, *i.e.*, **syntactic categories** (e.g. ‘WFF’)

*This will be of fundamental importance to the Montague Grammar architecture...*

Now, recall our formal definition of a ‘valuation’ for Propositional Logic, below:

(9) **Definition of a ‘Valuation’ for Propositional Logic**

A valuation  $V$  is a function from the well-formed formulae of PL to the set of truth-values  $\{1,0\}$  ( $V: \text{WFF} \rightarrow \{0,1\}$ ) such that if  $\varphi, \psi \in \text{WFF}$ , then

- a.  $V(\neg\varphi) = 1$  iff  $V(\varphi) = 0$
- b.  $V((\varphi \ \& \ \psi)) = 1$  iff  $V(\varphi) = 1$  and  $V(\psi) = 1$
- c.  $V((\varphi \vee \psi)) = 1$  iff  $V(\varphi) = 1$  or  $V(\psi) = 1$
- d.  $V((\varphi \rightarrow \psi)) = 1$  iff  $V(\varphi) = 0$  or  $V(\psi) = 1$

Again, just for fun – but with big implications for later – let’s introduce the following ‘semantic operations’:

(10) **Semantic Operations Over {1,0}**

The following operations (functions) take as input elements of either {1,0} or {1,0}<sup>2</sup>:

a. The Operation ‘Neg’

$$\text{Neg} = \{ <1,0>, <0,1> \}$$

b. The Operation ‘Conj’

$$\text{Conj} = \{ <<1,1>,1>, <<1,0>,0>, <<0,1>,0>, <<0,0>,0> \}$$

c. The Operation ‘Disj’

$$\text{Disj} = \{ <<1,1>,1>, <<1,0>,1>, <<0,1>,1>, <<0,0>,0> \}$$

d. The Operation ‘Imp’

$$\text{Imp} = \{ <<1,1>,1>, <<1,0>,0>, <<0,1>,1>, <<0,0>,1> \}$$

(11) **Observation:** {1,0} is closed under Neg, Conj, Disj, Imp

We can use the semantic operations in (10) and the syntactic operations in (3) to provide the following re-statement of what a valuation is...

(12) **Definition of a ‘Valuation’ for Propositional Logic**

A function  $V: \text{WFF} \rightarrow \{0,1\}$  is a valuation if it satisfies the conditions below:

a. If  $\varphi = \text{Not}(\psi)$ , then  $V(\varphi) = \text{Neg}(V(\psi))$

b. If  $\varphi = \text{And}(\psi, \chi)$ , then  $V(\varphi) = \text{Conj}(V(\psi), V(\chi))$

c. If  $\varphi = \text{Or}(\psi, \chi)$ , then  $V(\varphi) = \text{Disj}(V(\psi), V(\chi))$

d. If  $\varphi = \text{If}(\psi, \chi)$ , then  $V(\varphi) = \text{Imp}(V(\psi), V(\chi))$

Confirm For Yourself:

The new definition in (12) entails the key conditions of our earlier definition in (9):

a.  $V(\sim\varphi) = 1$  iff  $V(\varphi) = 0$

b.  $V((\varphi \& \psi)) = 1$  iff  $V(\varphi) = 1$  and  $V(\psi) = 1$

c.  $V((\varphi \vee \psi)) = 1$  iff  $V(\varphi) = 1$  or  $V(\psi) = 1$

d.  $V((\varphi \rightarrow \psi)) = 1$  iff  $V(\varphi) = 0$  or  $V(\psi) = 1$

## 2. Prolegomena, Part 2: A New Presentation of FOL

We'll also slightly alter our presentation of FOL in a manner similar to what we just did for PL...

- Recall our earlier definition below:

### (13) The Definition of a ‘Well-Formed Formula’ (WFF) in FOL

The set of ‘well-formed formulae’ of FOL, WFF, is the smallest set such that:

- If  $\varphi$  is an n-ary predicate letter and each of  $\alpha_1, \dots, \alpha_n$  is either an individual constant or a variable, then  $\varphi\alpha_1\dots\alpha_n \in \text{WFF}$
- If  $\varphi, \psi \in \text{WFF}$ , then
  - $\sim\varphi \in \text{WFF}$
  - $(\varphi \& \psi) \in \text{WFF}$
  - $(\varphi \vee \psi) \in \text{WFF}$
  - $(\varphi \rightarrow \psi) \in \text{WFF}$
- If  $\varphi \in \text{WFF}$  and  $v$  is a variable, then
  - $\forall v\varphi \in \text{WFF}$
  - $\exists v\varphi \in \text{WFF}$

For our first minor change, we'll slightly alter the syntax of the atomic formulae.

### (14) Slightly Altered Syntax for Atomic Formulae

If  $\varphi$  is an n-ary predicate letter and each of  $\alpha_1, \dots, \alpha_n$  is either an individual constant or a variable, then  $(\dots(\varphi\alpha_1)\dots\alpha_n) \in \text{WFF}$

Illustration:

Earlier Syntax:

Faxby

New Syntax:

$((\text{Fa})x)b)y$

*Now, let's expand the syntactic operations in (3) by adding the following...*

### (15) Syntactic Operations Over the Vocabulary of FOL

#### a. The Operation ‘Concat’

$\text{Concat}(\varphi, \psi) = \text{The result of concatenating ‘(’, } \varphi, \text{ and ‘)’}$

#### b. The Operation ‘All’

$\text{All}(\varphi, \psi) = \text{The result of concatenating ‘\forall’, } \varphi, \text{ and } \psi$

#### c. The Operation ‘Ext’

$\text{Ext}(\varphi, \psi) = \text{The result of concatenating ‘\exists’, } \varphi, \text{ and } \psi$

With these syntactic operations, we can restate our definition of WFF in FOL as follows:

**(16) The Definition of a ‘Well-Formed Formula’ (WFF) in FOL**

The set of ‘well-formed formulae’ of PL, WFF, is the smallest set such that:

- a. If  $\varphi$  is an n-ary predicate letter and each of  $\alpha_1, \dots, \alpha_n$  is either an individual constant or a variable, then  $\text{Concat}(\dots(\text{Concat}(\text{Concat}(\varphi, \alpha_1), \alpha_2), \dots, \alpha_n)) \in \text{WFF}$
- b. If  $\varphi \in \text{WFF}$ , then  $\text{Not}(\varphi) \in \text{WFF}$
- c. If  $\varphi, \psi \in \text{WFF}$ , then  $\text{And}(\varphi, \psi) \in \text{WFF}$
- d. If  $\varphi, \psi \in \text{WFF}$ , then  $\text{Or}(\varphi, \psi) \in \text{WFF}$
- e. If  $\varphi, \psi \in \text{WFF}$ , then  $\text{If}(\varphi, \psi) \in \text{WFF}$
- f. If  $\varphi \in \text{WFF}$  and  $v$  is a variable, then  $\text{Ext}(v, \varphi) \in \text{WFF}$
- g. If  $\varphi \in \text{WFF}$  and  $v$  is a variable, then  $\text{All}(v, \varphi) \in \text{WFF}$

We could also use our schematic notation in (7) to provide the following even more compact presentation of the rules in (16b)-(16g).<sup>2</sup>

**(17) Abstract Representation of Rules (16b)-(16g)**

- b.  $\langle \text{Not}, \langle \text{WFF} \rangle, \text{WFF} \rangle$   
‘The result of applying Not to a member of WFF is a WFF’
- c.  $\langle \text{And}, \langle \text{WFF}, \text{WFF} \rangle, \text{WFF} \rangle$   
‘The result of applying And to a pair consisting of a WFF and a WFF is a WFF’
- d.  $\langle \text{Or}, \langle \text{WFF}, \text{WFF} \rangle, \text{WFF} \rangle$   
‘The result of applying Or to a pair consisting of a WFF and a WFF is a WFF’
- e.  $\langle \text{If}, \langle \text{WFF}, \text{WFF} \rangle, \text{WFF} \rangle$   
‘The result of applying If to a pair consisting of a WFF and a WFF is a WFF’
- f.  $\langle \text{Ext}, \langle \text{VAR}, \text{WFF} \rangle, \text{WFF} \rangle$   
‘The result of applying Ext to a pair consisting of a VAR and a WFF is a WFF’
- g.  $\langle \text{All}, \langle \text{VAR}, \text{WFF} \rangle, \text{WFF} \rangle$   
‘The result of applying All to a pair consisting of a VAR and a WFF is a WFF’

---

<sup>2</sup> Rule (16a) won’t be schematically representable in this way until we’ve made one last change, to come later.

Given our new notation for atomic formulae and our semantic operations in (10), we'll make a slight change to our definitions of 'model' and 'valuation w.r.t. a model and variable assignment'

### (18) Definition of a 'Model' for First Order Logic

A model  $\mathcal{M}$  is a pair  $\langle D, I \rangle$  consisting of:

- a. A *non-empty* set  $D$ , called the 'domain of  $\mathcal{M}$ '
- b. A function  $I$ , whose domain is the individual constants and predicate letters, and whose range satisfies the following conditions:
  - (i) If  $\alpha$  is an individual constant, then  $I(\alpha) \in D$
  - (ii) If  $\Phi$  is an  $n$ -ary predicate letter, then  $I(\Phi)$  is **the curried characteristic function of an  $n$ -ary relation  $R \subseteq D^n$**

Note:

We are now interpreting predicate letters not as subsets of  $D$  or relations on  $D$ , but as **the curried characteristic functions** of such sets and relations.

### (19) Valuation of FOL, Relative to a Model and a Variable Assignment

Let  $\mathcal{M}$  be a model  $\langle D, I \rangle$  and  $g$  be a variable assignment (based on  $\mathcal{M}$ ). Then the 'valuation based on  $M$  and  $g$ ' ( $V_{M,g}$ ) is a function whose domain is the set of FOL formulae, whose range is  $\{0,1\}$ , and which satisfies the conditions below:

- (i) If  $\varphi = \text{Concat}(\dots(\text{Concat}(\Phi, \alpha_1), \dots, \alpha_n), \dots)$ , then  $V_{M,g}(\varphi) = I(\Phi)([[\alpha_1]]^{M,g}) \dots ([[\alpha_n]]^{M,g})$ <sup>3</sup>
- (ii) If  $\varphi = \text{Not}(\psi)$ , then  $V_{M,g}(\varphi) = \text{Neg}(V_{M,g}(\psi))$
- (iii) If  $\varphi = \text{And}(\psi, \chi)$ , then  $V_{M,g}(\varphi) = \text{Conj}(V_{M,g}(\psi), V_{M,g}(\chi))$
- (iv) If  $\varphi = \text{Or}(\psi, \chi)$ , then  $V_{M,g}(\varphi) = \text{Disj}(V_{M,g}(\psi), V_{M,g}(\chi))$
- (v) If  $\varphi = \text{If}(\psi, \chi)$ , then  $V_{M,g}(\varphi) = \text{Imp}(V_{M,g}(\psi), V_{M,g}(\chi))$
- (vi) If  $\varphi = \text{Ext}(v, \psi)$ , then  $V_{M,g}(\varphi) = 1$  iff there is an  $a \in D$  such that  $V_{M,g(v/a)}(\psi) = 1$
- (vii) If  $\varphi = \text{All}(v, \psi)$ , then  $V_{M,g}(\varphi) = 1$  iff for every  $a \in D$ ,  $V_{M,g(v/a)}(\psi) = 1$

---

<sup>3</sup> Note that if  $I(\Phi)$  is the curried characteristic function of  $R$ , then this is equivalent to saying  $\langle [[\alpha_1]]^{M,g}, \dots, [[\alpha_n]]^{M,g} \rangle \in R$ , as in our original definition of a valuation based on  $M$  and  $g$ .

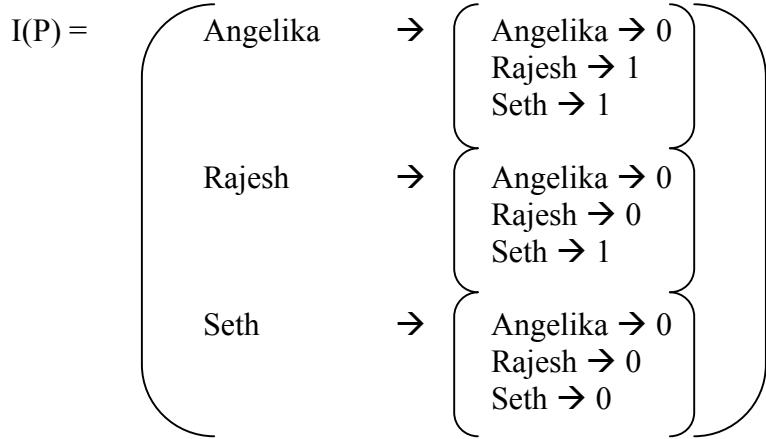
(20) **Illustration of the New Definition**

a. Illustration of the Definition of a Model:

The pair  $\text{FOL} < \{ \text{Angelika}, \text{Seth}, \text{Rajesh} \}, I >$  is a model of FOL, where  $I$  consists of at least the following mappings:<sup>4</sup>

$$I(a) = \text{Angelika} \quad I(b) = \text{Seth} \quad I(c) = \text{Rajesh}$$

$$I(F) = \{ <\text{Angelika}, 0>, <\text{Seth}, 0>, <\text{Rajesh}, 0> \}$$



b. Illustration of the Definition of a Valuation:

Let  $\mathcal{M}$  be the model (partially) defined in (20a) and  $g$  be any variable assignment based on  $\mathcal{M}$

- $V_{M,g}( \sim((Pb)c) ) = \text{ (by definition of Not and Concat)}$
- $V_{M,g}( \text{Not( Concat( Concat(P,b), c ) ) } = \text{ (by (19ii))}$
- $\text{Neg}( V_{M,g}( \text{Concat( Concat(P,b), c ) ) } = \text{ (by (19i))}$
- $\text{Neg}( I(P)([b]^{M,g})([c]^{M,g}) ) = \text{ (by definition of } [[.]]^{M,g}\text{)}$
- $\text{Neg}( I(P)(I(b))(I(c)) ) = \text{ (by definition of I)}$
- $\text{Neg}( I(P)(\text{Seth})(\text{Rajesh}) ) = \text{ (by definition of I(P))}$
- $\text{Neg}( 0 ) = \text{ (by definition of Neg)}$
- $1$

---

<sup>4</sup> Note that we are again basically interpreting ‘F’ as “is French” and ‘P’ as “is older than”.

### 3. Algebras and Morphisms: The Key Concepts

Various advances in algebra in the 18<sup>th</sup> and 19<sup>th</sup> century lead mathematicians to develop a highly general and abstract definition of what a system of ‘algebra’ is.

#### (21) Definition of an Algebra

An *algebra* is a tuple  $\langle A, f_1, \dots, f_n \rangle$  consisting of a set  $A$  together with one or more operations (functions)  $f_1, \dots, f_n$ , where  $A$  is *closed* under each of  $f_1, \dots, f_n$ .

- Note: The operations  $f_1, \dots, f_n$  don’t have to be of the same arity, but they must be of some finite arity.

#### (22) Illustrations of Algebras

a.  $\langle \mathbb{N}, +, \times \rangle$

- The natural numbers are *closed* under addition and multiplication

Note: This is not an algebra:  $\langle \mathbb{N}, +, \times, - \rangle$ , since  $\mathbb{N}$  isn’t closed under subtraction

b.  $\langle \mathbb{Z}, +, \times, - \rangle$

- The integers are *closed* under addition, multiplication, and subtraction

Note: This is not an algebra:  $\langle \mathbb{Z}, +, \times, -, \div \rangle$ , since  $\mathbb{Z}$  isn’t closed under division.

c.  $\langle \mathbb{Q}, +, \times, -, \div \rangle$

- The rationals are *closed* under addition, multiplication, subtraction, division

d.  $\langle \text{WFF}_{\text{PL}}, \text{Not}, \text{And}, \text{Or}, \text{If} \rangle$

- The WFF of PL are *closed* under Not, And, Or, and If (see (6))

e.  $\langle \{1,0\}, \text{Neg}, \text{Conj}, \text{Disj}, \text{Imp} \rangle$

- The set  $\{1,0\}$  is *closed* under Neg, Conj, Disj, Imp (see (11))

f.  $\langle \{1,0\}, \text{Conj}, \text{Disj} \rangle$

g.  $\langle \{\{a\}, \emptyset\}, \cap, \cup \rangle$

- The set  $\{\{a\}, \emptyset\}$  is closed under intersection and union

$$\{a\} \cup \{a\} = \{a\}$$

$$\{a\} \cap \{a\} = \{a\}$$

$$\{a\} \cup \emptyset = \{a\}$$

$$\{a\} \cap \emptyset = \emptyset$$

$$\emptyset \cup \{a\} = \{a\}$$

$$\emptyset \cap \{a\} = \emptyset$$

$$\emptyset \cup \emptyset = \emptyset$$

$$\emptyset \cap \emptyset = \emptyset$$

(23) **Key Observation**

The algebras  $\langle \{1,0\}, \text{Conj}, \text{Disj} \rangle$  and  $\langle \{\{a\}, \emptyset\}, \cap, \cup \rangle$  are intuitively ‘similar’

$\langle \{1,0\}, \text{Conj}, \text{Disj} \rangle$	$\langle \{\{a\}, \emptyset\}, \cap, \cup \rangle$
$\text{Conj}(1,1) = 1$	$\cap(\{a\}, \{a\}) = \{a\}$
$\text{Conj}(1,0) = 0$	$\cap(\{a\}, \emptyset) = \emptyset$
$\text{Conj}(0,1) = 0$	$\cap(\emptyset, \{a\}) = \emptyset$
$\text{Conj}(0,0) = 0$	$\cap(\emptyset, \emptyset) = \emptyset$
$\text{Disj}(1,1) = 1$	$\cup(\{a\}, \{a\}) = \{a\}$
$\text{Disj}(1,0) = 1$	$\cup(\{a\}, \emptyset) = \{a\}$
$\text{Disj}(0,1) = 1$	$\cup(\emptyset, \{a\}) = \{a\}$
$\text{Disj}(0,0) = 0$	$\cup(\emptyset, \emptyset) = \emptyset$

This intuitive notion of ‘similarity’ is captured in the following notion of ‘isomorphism’.

(24) **Isomorphic and Isomorphism**

Let  $\mathbf{A} \langle A, f_1, \dots, f_n \rangle$  and  $\mathbf{B} \langle B, g_1, \dots, g_n \rangle$  be algebras. We say that  $\mathbf{A}$  and  $\mathbf{B}$  are *isomorphic* if the following hold:

- (i) There is a one-to-one correspondence between the operations  $f_1, \dots, f_n$  and  $g_1, \dots, g_n$
- (ii) Each  $f_i$  is of the same arity as  $g_i$
- (iii) There is a bijection  $h: A \rightarrow B$  with the following key property:

$$\text{For every operation } f_i \text{ in } \mathbf{A}, h(f_i(a_1, \dots, a_m)) = g_i(h(a_1), \dots, h(a_n))$$

If  $\mathbf{A}$  and  $\mathbf{B}$  are isomorphic, then the bijection  $h: A \rightarrow B$  is an *isomorphism*.

Note: Informally, (iii) states that that:

If you apply bijection  $h$  to the result of  $f_i$  applied to  $a_1$  through  $a_m$ , you get the same thing in  $B$  as you’d get if

You applied the bijection  $h$  to  $a_1$  through  $a_m$  to get a sequence  $b_1$  through  $b_m$

You then applied the corresponding function  $g_i$  in  $\mathbf{B}$  to  $b_1$  through  $b_m$

Thus, if  $f_i(a_1, \dots, a_m) = a$ , then  $g_i(h(a_1), \dots, h(a_n)) = h(a)$

- And so we see how the isomorphism  $h$  entails that the structure of the algebra  $\mathbf{A}$  is ‘mirrored’ in the algebra  $\mathbf{B}$

(25) **Illustration**

The algebras  $\langle \{1,0\}, \text{Conj}, \text{Disj} \rangle$  and  $\langle \{\{a\}, \emptyset\}, \cap, \cup \rangle$  are isomorphic.

Demonstration: Assume the correspondence  $\text{Conj} \sim \cap$  and  $\text{Disj} \sim \cup$   
Consider the bijection  $h = \{ \langle 1, \{a\} \rangle, \langle 0, \emptyset \rangle \}$

*Checking For*  $\text{Conj} \sim \cap$

$$\begin{aligned} h(\text{Conj}(1,1)) &= h(1) = \{a\} = \cap(\{a\}, \{a\}) = \cap(h(1), h(1)) \\ h(\text{Conj}(1,0)) &= h(0) = \emptyset = \cap(\{a\}, \emptyset) = \cap(h(1), h(0)) \\ h(\text{Conj}(0,1)) &= h(0) = \emptyset = \cap(\emptyset, \{a\}) = \cap(h(0), h(1)) \\ h(\text{Conj}(0,0)) &= h(0) = \emptyset = \cap(\emptyset, \emptyset) = \cap(h(0), h(0)) \end{aligned}$$

*Checking For*  $\text{Disj} \sim \cup$

$$\begin{aligned} h(\text{Disj}(1,1)) &= h(1) = \{a\} = \cup(\{a\}, \{a\}) = \cup(h(1), h(1)) \\ h(\text{Disj}(1,0)) &= h(1) = \{a\} = \cup(\{a\}, \emptyset) = \cup(h(1), h(0)) \\ h(\text{Disj}(0,1)) &= h(1) = \{a\} = \cup(\emptyset, \{a\}) = \cup(h(0), h(1)) \\ h(\text{Disj}(0,0)) &= h(0) = \emptyset = \cup(\emptyset, \emptyset) = \cup(h(0), h(0)) \end{aligned}$$

Thus, the function  $h: A \rightarrow B$  is an *isomorphism* from **A** to **B**.

In addition to the notion of ‘isomorphism’ in (24), there’s also a weaker notion of ‘similarity’ between algebras, that of ‘homomorphism’.

(26) **Homomorphic and Homomorphism**

Let **A**  $\langle A, f_1, \dots, f_n \rangle$  and **B**  $\langle B, g_1, \dots, g_n \rangle$  be algebras. We say that **A** and **B** are *homomorphic* if the following hold:

- (i) There is a one-to-one correspondence between the operations  $f_1, \dots, f_n$  and  $g_1, \dots, g_n$
- (ii) Each  $f_i$  is of the same arity as  $g_i$
- (iii) There is a function  $h: A \rightarrow B$  with the following key property:

For every operation  $f_i$  in **A**,  $h(f_i(a_1, \dots, a_m)) = g_i(h(a_1), \dots, h(a_n))$

If **A** and **B** are homomorphic, then the bijection  $h: A \rightarrow B$  is an *homomorphism*.

Note:

A homomorphism differs from an isomorphism in that the former need not be an injection or a surjection; it can map two different things in **A** to the same thing in **B**, or there could be things in **B** that nothing in **A** gets mapped to...

(27) **Key Illustration**

The algebras  $\langle \text{WFF}_{\text{PL}}, \text{Not}, \text{And}, \text{Or}, \text{If} \rangle$  and  $\langle \{1,0\}, \text{Neg}, \text{Conj}, \text{Disj}, \text{Imp} \rangle$  are homomorphic.

Demonstration:

Assume the correspondence  $\text{Not} \sim \text{Neg}$ ,  $\text{And} \sim \text{Conj}$ ,  $\text{Or} \sim \text{Disj}$ , and  $\text{If} \sim \text{Imp}$ .  
Let  $V$  be *any valuation*, as defined in (12).

- o Note,  $V$  is a function from  $\text{WFF}_{\text{PL}}$  to  $\{1,0\}$ .
- o Thus, we need only check the key correspondence property in (26iii)

$$\text{Checking For Not} \sim \text{Neg}: \quad V(\text{Not}(\psi)) = \text{Neg}(V(\psi)) \quad (\text{by (12a)})$$

$$\text{Checking For And} \sim \text{Conj}: \quad V(\text{And}(\psi, \chi)) = \text{Conj}(V(\psi), V(\chi)) \quad (\text{by (12b)})$$

$$\text{Checking For Or} \sim \text{Disj}: \quad V(\text{Or}(\psi, \chi)) = \text{Disj}(V(\psi), V(\chi)) \quad (\text{by (12c)})$$

$$\text{Checking for If} \sim \text{Imp}: \quad V(\text{If}(\psi, \chi)) = \text{Imp}(V(\psi), V(\chi)) \quad (\text{by (12c)})$$

---

#### 4. Semantics and Homomorphism

(28) **Key Consequence of (27)**

A valuation  $V: \text{WFF}_{\text{PL}} \rightarrow \{1,0\}$  is a homomorphism from  $\langle \text{WFF}_{\text{PL}}, \text{Not}, \text{And}, \text{Or}, \text{If} \rangle$  ('syntactic algebra') to  $\langle \{1,0\}, \text{Neg}, \text{Conj}, \text{Disj}, \text{Imp} \rangle$  ('semantic algebra')

*One of the crucial insights of Montague was to generalize from (28) in the following way...*

(29) **An 'Interpretation' of Propositional Logic (To Be Revised)**

Let the language of PL be the algebra  $L = \langle \text{WFF}_{\text{PL}}, \text{Not}, \text{And}, \text{Or}, \text{If} \rangle$ . An *interpretation* of PL is a structure  $\langle B, f_1, f_2, f_3, f_4, h \rangle$  such that:

- a.  $\langle B, f_1, f_2, f_3, f_4 \rangle$  is an algebra
- b.  $h$  is a homomorphism from  $L$  to  $\langle B, f_1, f_2, f_3, f_4 \rangle$

(30) **Key Consequence:**

If  $V$  is a valuation, then  $\langle \{1,0\}, \text{Neg}, \text{Conj}, \text{Disj}, \text{Imp}, V \rangle$  is an interpretation of PL.

*However, as shown below, the notion of 'interpretation' is more general than that of 'valuation'*

(31) **An Interpretation That is Not Based on Valuations**

Let  $h: \text{WFF}_{\text{PL}} \rightarrow P(\{a,b,c\})^5$  be a function from  $\text{WFF}_{\text{PL}}$  to the powerset of  $\{a,b,c\}$  with the following key properties:

- a.  $h(\neg\varphi) = h(\text{Not}(\varphi)) = h(\varphi)'$
- b.  $h((\varphi \& \psi)) = h(\text{And}(\varphi, \psi)) = \cap(h(\varphi), h(\psi))$
- c.  $h((\varphi \vee \psi)) = h(\text{Or}(\varphi, \psi)) = \cup(h(\varphi), h(\psi))$
- d.  $h((\varphi \rightarrow \psi)) = h(\text{If}(\varphi, \psi)) = \text{IMP}(h(\varphi), h(\psi))$   
  - Where  $\text{IMP}(A,B) = A' \cup B$

Illustration:

If  $h(p) = \{b,c\}$  and  $h(q) = \{a,b\}$ , then  $h((p \& q)) = \{b\}$ , and  $h((p \vee q)) = \{a,b,c\}$

Key Consequence:

The structure  $\langle P(\{a,b,c\}), ', \cap, \cup, \text{IMP}, h \rangle$  is an interpretation of PL.

- $\langle P(\{a,b,c\}), ', \cap, \cup, \text{IMP} \rangle$  is an algebra
- $h$  is a homomorphism from  $L$  to  $\langle P(\{a,b,c\}), ', \cap, \cup, \text{IMP} \rangle$ 
  - $h$  is a function from  $\text{WFF}_{\text{PL}}$  to  $P(\{a,b,c\})$
  - The assumptions in (31a-d) show that it has the homomorphism property (26iii)

We could view an interpretation like  $\langle P(\{a,b,c\}), ', \cap, \cup, \text{IMP}, h \rangle$  as being one in which the WFFs of PL are mapped to the ‘possible worlds’ in  $\{a,b,c\}$  where they are true...

*Finally, we can (try to) generalize the notion of ‘interpretation’ in (29) to any language...*

(32) **Generalizing ‘Interpretation’ for Any Language (To Be Revised)**

Let  $L = \langle A, f_1, \dots, f_n \rangle$  be a ‘syntactic algebra’ for a given language. An *interpretation* of  $L$  is a structure  $\langle B, g_1, \dots, g_n, j \rangle$  such that:

- a.  $\langle B, g_1, \dots, g_n \rangle$  is an algebra.
- b. For any  $i$ ,  $f_i$  and  $g_i$  have the same arity.
- c.  $j$  is a function from the ‘lexical items’ (e.g. ‘non-logical constants’) in  $A$  to  $B$ .

If  $\mathbf{B} = \langle B, g_1, \dots, g_n, j \rangle$  is an interpretation of  $L$ , then the *meaning assignment* for  $L$  determined by  $\mathbf{B}$  is the unique homomorphism  $h$  from  $L$  to  $\langle B, g_1, \dots, g_n \rangle$  such that  $j \subseteq h$ .

---

<sup>5</sup> That such a function exists can be shown by giving a concrete example. Students familiar with modal propositional logic will be familiar with many such concrete examples.

(33) **Semantics and Homomorphism: The Intuitive Relation**

a. The Principle of Compositionality

The meaning of a complex expression is a ‘function of’ (i) the meanings of its component expressions and (ii) their mode of syntactic composition.

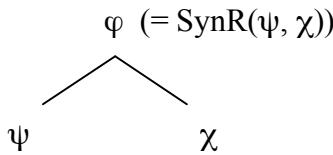
b. Key Observation:

(33a) basically says that interpretation is homomorphism from syntax to meaning

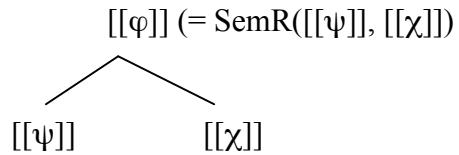
- To get the meaning of a complex expression  $\varphi$ , you determine the syntactic rule SynR that derives it from its component expressions  $\psi, \chi$
- Then, you compute the meanings  $[[\psi]], [[\chi]]$
- Finally, you input  $[[\psi]], [[\chi]]$  into a semantic rule SemR that ‘corresponds’ to the syntactic rule SynR

$$\circ \quad \text{Thus, } [[\text{SynR}(\psi, \chi)]] = \text{SemR}([[\psi]], [[\chi]])$$

*Syntax:*



*Semantics*



(34) **Semantics and Homomorphism: Another Possible Connection**

Suppose that interpretation is homomorphism from syntax to meaning.

- It follows that if a language has a semantics, there is a ‘similarity’ (homomorphism) between its structure (syntactic algebra) and mind-external reality (semantic algebra)

a. Picture Theory of Meaning (Early Wittgenstein):

A meaningful expression ‘pictures’ a state of affairs / atomic fact.

(There is a structural similarity between the structure of the expression and the ‘logical facts’ of the world.)

b. Correspondence Theory of Truth (Russell, et multia alia)

For a statement to be true, it must have a structural isomorphism with the state of affairs that makes it true.

“‘A cat is on the mat’ is true iff there is in the world a cat and a mat, and the cat is related to the mat in virtue of being on it. If any of the three pieces is missing... then the statement is false.”

## 5. Moving Beyond Propositional Logic: The Key Challenges

If the definition in (32) is truly viable, then all we need to do to provide a semantics for a language  $L$  is the following:

- Characterize the language  $L$  as a ‘syntactic algebra’  $\langle A, f_1, \dots, f_n \rangle$
- Find a structure  $\langle B, g_1, \dots, g_n, j \rangle$  that satisfies the key properties of being an ‘interpretation’ of the language.

Thus, in order for the general semantic program embodied in (32) to be viable, we must be able to characterize every (semantically interpreted) language as a syntactic algebra...

### (35) Key Problem: First Order Logic!

- Given our presentation in (16), the key syntactic operations forming the WFFs of FOL are: Concat, Not, And, Or, If, Ext, All.
- However, the structure  $\langle \text{WFF}_{\text{FOL}}, \text{Concat}, \text{Not}, \text{And}, \text{Or}, \text{If}, \text{Ext}, \text{All} \rangle$  is *not* an algebra!
  - $\text{WFF}_{\text{FOL}}$  is *not* closed under Concat, Ext, and All
    - $\text{Concat}( \text{Pa} , ((\text{Rb})\text{c}) ) = (\text{Pa}((\text{Rb})\text{c})) \notin \text{WFF}_{\text{FOL}}$
    - $\text{Ext}( \text{Pa}, ((\text{Rb})\text{c}) ) = \exists \text{Pa}((\text{Rb})\text{c}) \notin \text{WFF}_{\text{FOL}}$
    - $\text{All}( \text{Pa}, ((\text{Rb})\text{c}) ) = \forall \text{Pa}((\text{Rb})\text{c}) \notin \text{WFF}_{\text{FOL}}$

#### Main Issue:

In PL, applying any of the syntactic operations to a WFF creates a WFF. This just isn’t the case for FOL

#### The Solution:

We’ll need to augment what we mean by an ‘algebraic characterization’ of a language, and thus the definition in (32).

### (36) Another Challenge Raised by FOL

Even once we resolve the issue in (35), to pursue the general program outlined in (32), we’ll need to find some ‘semantic operations’ that ‘correspond’ to Ext and All

- Finding such operations is what logicians mean by ‘algebraization of FOL’.
- Doing this is far from trivial; Montague’s solution is not commonly used and difficult

### Problem Set on Syntactic Operations, Semantic Operations, and Homomorphisms

#### (1) Exercise on the Syntactic Operations for PL

Please show how the following formulae of PL can be constructed via applications of the syntactic operations Not, And, Or, and If.

For example, if the formula were  $(p \rightarrow \neg(q \ \& \ r))$ , the correct answer would be  $\text{If}(p, \text{Not}(\text{And}(q, r)))$ .

- a.  $\neg(p \ \& \ (p \rightarrow (q \vee r)))$
- b.  $((p \vee r) \rightarrow q) \vee s$
- c.  $((p \rightarrow q) \rightarrow (\neg q \rightarrow \neg p)) \ \& \ ((p \rightarrow q) \rightarrow (\neg p \vee q))$

#### (2) Exercise on the Syntactic Operations for FOL

Please show how the following formulae of FOL can be constructed via applications of the syntactic operations Concat, Not, And, Or, If, Ext, and All.

For example, if the formula were  $(\exists x((Px)a) \ \& \ (Rb))$ , the correct answer would be  $\text{And}(\text{Ext}(x, \text{Concat}(\text{Concat}(P, x), a)), \text{Concat}(R, b))$ .

- a.  $\forall x((Px) \rightarrow \exists y((Lx)y))$
- b.  $\exists z(((Ra)z) \vee \neg((Ra)z))$
- c.  $((Ab) \vee \neg\exists x(Ax))$

#### (3) Exercise on (Montagovian) Syntactic Rules

Let Merge be an operation that takes two strings  $\varphi$  and  $\psi$  and forms the string ' $\varphi \ \psi$ '. Using this operation Merge, please show how each of the PS rules below can be abstractly characterized as triple  $\langle \text{Op}, \langle \text{Cat}_1, \dots, \text{Cat}_n \rangle, \text{Cat} \rangle$ , where Op is an  $n$ -ary syntactic operation, and each of  $\text{Cat}_i$  and Cat is some syntactic category label.

- a.  $\text{DP} \rightarrow \text{D NP}$
- b.  $\text{PP} \rightarrow \text{P NP}$
- c.  $\text{NP} \rightarrow \text{A NP}$
- d.  $\text{S} \rightarrow \text{NP VP}$
- e. How could we capture the following PS rule using this 'triplet' notation? Please do not in any way alter the notation:  
$$\text{VP} \rightarrow \text{V} (\text{NP})$$

**(4) Our New Definition of a Model**

Let  $\mathcal{M}$  be a model  $\langle D, I \rangle$  as defined in the handout “First Order Logic: Formal Semantics and Models”, where  $D = \{ a, b, c \}$ , and  $I$  consists of at least the mappings below:

$$I(P) = \{ x : x \in D \text{ and } x \text{ is a vowel} \}$$

$$I(Q) = \{ \langle x, y \rangle : x, y \in D \text{ and } x \text{ precedes } y \text{ in the alphabet} \}$$

- a. Please convert  $\mathcal{M}$  into a model as defined in (18) on the handout “An Algebraic Perspective on Propositional Logic.” That is, state what  $D$  and  $I$  should be under the new definition in (18).
- b. Please use the new definition of ‘valuation’ in (19) on the handout “An Algebraic Perspective on Propositional Logic” to show how the converted model  $\mathcal{M}$  assigns truth-values to the following formulae.
  - (i)  $\exists x(Px)$
  - (ii)  $\forall x \exists y((Qy)x)$

**(5) Exercise on ‘Interpretations’ for PL**

Let  $\langle \{0,1\}, \text{Neg}, \text{Conj}, \text{Disj}, \text{Imp}, V \rangle$  be an interpretation for PL, as defined in (29) of “An Algebraic Perspective on Propositional Logic.” Moreover, assume that  $V$  is such that  $V(p) = 1$ ,  $V(q) = 0$ , and  $V(r) = 1$ . Please use the assumption that  $V$  is a homomorphism to calculate truth-values of the following formulae.

For example, if the formula were  $(p \rightarrow q)$ , then the calculation would be:

- (i)  $V(p \rightarrow q) =$  (by definition of PL)
- (ii)  $V(\text{If}(p,q)) =$  (by homomorphism property of  $V$ )
- (iii)  $\text{Imp}(V(p), V(q)) =$  (by definition of  $V$ )
- (iv)  $\text{Imp}(1, 0) =$  (by definition of Imp)
- (v)  $0$

- a.  $\sim(p \& (p \rightarrow (q \vee r)))$
- b.  $((p \vee \sim r) \rightarrow q) \vee p$
- c.  $((p \rightarrow q) \rightarrow (\sim q \rightarrow \sim p)) \& ((p \rightarrow q) \rightarrow (\sim p \vee q))$

**(6) Exercise on Homomorphisms and Compositions**

Let  $\mathbf{A} \langle A, f_1, \dots, f_n \rangle$  and  $\mathbf{B} \langle B, g_1, \dots, g_n \rangle$  and  $\mathbf{C} \langle C, h_1, \dots, h_n \rangle$  all be algebras. Assume that  $k: \mathbf{A} \rightarrow \mathbf{B}$  is a homomorphism from  $\mathbf{A}$  to  $\mathbf{B}$ , and that  $j: \mathbf{B} \rightarrow \mathbf{C}$  is a homomorphism from  $\mathbf{B}$  to  $\mathbf{C}$ . Please show that  $j \circ k$  is a homomorphism from  $\mathbf{A}$  to  $\mathbf{C}$ .

Note: To properly show this, you need to show the following:

- (i) That  $j \circ k$  is a function from  $A$  to  $C$  [trivial]
- (ii) That the operations  $f_1, \dots, f_n$  can be put into a one-to-one correspondence with  $h_1, \dots, h_n$ , such that for all  $i \in \mathbb{N}$ ,  $f_i$  is the same arity as  $h_i$  [trivial]
- (iii) That for all  $i \in \mathbb{N}$ ,  $j \circ k(f_i(a_1, \dots, a_m)) = h_i(j \circ k(a_1), \dots, j \circ k(a_m))$  [important part]

## An Algebraic Perspective on the Syntax of First Order Logic (Without Quantification)<sup>1</sup>

### 1. Statement of the Problem, Outline of the Solution to Come

#### (1) The Key Problem

- There is much to recommend an ‘algebraic’ treatment of interpretation, where it is conceived of as a homomorphism from a ‘syntactic’ algebra to a ‘semantic’ one.
- Based on our algebraic semantics for PL, we were lead to the (preliminary) generalization in (2) below.
- However, its not possible to directly extend this conception of interpretation to FOL:
  - Unlike PL, the structure  $\langle \text{WFF}_{\text{FOL}}, \text{Concat}, \text{Not}, \text{And}, \text{Or}, \text{If}, \text{Ext}, \text{All} \rangle$  is *not* an algebra, because **WFF<sub>FOL</sub> isn't closed under those operations...**

#### (2) Algebraic Perspective on ‘Interpretation’ (To Be Revised)

Let  $L = \langle A, f_1, \dots, f_n \rangle$  be a ‘syntactic algebra’ for a given language. An *interpretation* of  $L$  is a structure  $\langle B, g_1, \dots, g_n, j \rangle$  such that:

- a.  $\langle B, g_1, \dots, g_n \rangle$  is an algebra.
- b. For any  $i$ ,  $f_i$  and  $g_i$  have the same arity.
- c.  $j$  is a function from the ‘non-logical constants’ in  $A$  to  $B$ .

If  $\mathbf{B} = \langle B, g_1, \dots, g_n, j \rangle$  is an interpretation of  $L$ , then the *meaning assignment* for  $L$  determined by  $\mathbf{B}$  is the unique homomorphism  $h$  from  $L$  to  $\langle B, g_1, \dots, g_n \rangle$  such that  $j \subseteq h$ .

#### (3) General Outline of a Solution

We’re going to loosen what it means to be a ‘syntactic algebra’ for a given language.

- In a ‘syntactic algebra’ for  $L$ ,  $\langle A, f_1, \dots, f_n \rangle$ , the set  $A$  can be a *strict superset* of the well-formed sentences of the language.
- For FOL, we’ll construct an algebra  $\langle A, \text{Concat}, \text{Not}, \text{And}, \text{Or}, \text{If}, \text{Ext}, \text{All} \rangle$  such that  $\text{WFF}_{\text{FOL}} \subset A$ .
- We’ll then see how we can define/characterize  $\text{WFF}_{\text{FOL}}$  in terms of this algebra (with a few added formal ingredients)
- **In the next notes, we’ll build an algebraic semantics for  $\text{WFF}_{\text{FOL}}$  from these tools**

<sup>1</sup> These notes are based upon material in the following readings: Halvorsen & Ladusaw (1979), Dowty *et al.* (1981) Chapter 8, and Thomason (1974) Chapter 7 (Montague’s “Universal Grammar”).

#### (4) Zooming In: Obtaining WFF<sub>FOL</sub> From Our Syntactic Algebra

We'll construct WFF<sub>FOL</sub> from < A, Concat, Not, And, Or, If, Ext, All > by using a 'generate and filter' approach not too unlike classic GB:

- In the algebra  $\langle A, \text{Concat}, \text{Not}, \text{And}, \text{Or}, \text{If}, \text{Ext}, \text{All} \rangle$ ,  $A$  will contain all the WFF's of FOL, *but also a whole bunch of 'syntactic garbage'* (e.g.,  $(P \& a)$ ,  $\sim x$ , etc.)
  - We'll then show how syntactic rules function as a kind of *filter* over  $A$  (in the GB sense), separating out the WFFs from the 'syntactic garbage'.
  - We'll use these rules ('syntactic filters') to formally define that subset of  $A$  which is the well-formed formulae of FOL.
  - In the next notes, we'll make a minimal change to (2) so that it will straightforwardly extend to the complex system consisting of the algebra and these 'filters'.
  - **Funky Property of Our Resulting Semantics:**
    - We'll still keep the idea that an interpretation of FOL is a homomorphism from  $\langle A, \text{Concat}, \text{Not}, \text{And}, \text{Or}, \text{If}, \text{Ext}, \text{All} \rangle$  to a 'semantic algebra'.
    - **Thus, our 'algebraic semantics' will interpret – not only the WFFs of FOL – but also all the syntactic garbage (!!?)**

*As always, though, to make our lives easier, we'll start out by considering only the set of sentences of FOL that don't have any quantifiers.*

We'll also assume a minimal set of sentence connectives ( $\&$ ,  $\sim$ )...

### (5) Fragment of FOL We Will Focus on For Now: ‘FOL-NoQ’

The set of well-formed formulae of FOL-NoQ,  $\text{WFF}_{\text{FOL-NoQ}}$ , is the smallest set such that:

- a. If  $\varphi$  is an n-ary predicate letter and each of  $\alpha_1, \dots, \alpha_n$  is an individual constant, then  $\text{Concat}(\dots(\text{Concat}(\text{Concat}(\varphi, \alpha_1), \alpha_2), \dots, \alpha_n)) \in \text{WFF}_{\text{FOL-NoQ}}$
  - b. If  $\varphi \in \text{WFF}$ , then  $\text{Not}(\varphi) \in \text{WFF}$
  - c. If  $\varphi, \psi \in \text{WFF}$ , then  $\text{And}(\varphi, \psi) \in \text{WFF}$

<u>Examples of WFF of FOL-NoQ:</u>	$((Qa)b) \& \sim(Rc)$
	$\sim(\sim(Ab) \& \sim((Sc)d)) \approx ((Ab) \vee ((Sc)d))$
	$\sim(((Tg)b) \& \sim(Lg)) \approx (((Tg)b) \rightarrow (Lg))$

## 2. An Algebraic Characterization of $\text{WFF}_{\text{FOL-NoQ}}$

In this section, we're going to show how the set  $\text{WFF}_{\text{FOL-NoQ}}$  can be characterized in terms of a syntactic algebra. There will be six major steps:

1. Building the **Syntactic Algebra**
2. Introducing the **Syntactic Category Labels** (*i.e.*, the Types)
3. Using the Category Labels to Define the '**Basic Expressions**' of FOL-NoQ (*i.e.*, the non-logical constants)
4. Introducing the **Syntactic Rules** of FOL-NoQ
5. Using the Syntactic Rules and the Basic Expressions (non-logical constants) to Obtain the **Syntactic Categories** of FOL-NoQ
6. Putting It All Together: Defining  $\text{WFF}_{\text{FOL-NoQ}}$  in Terms of these Ingredients

Note:

As we'll see, the resulting system generating  $\text{WFF}_{\text{FOL-NoQ}}$  can be viewed as a kind of abstract mathematical characterization of 'what's going on' in recursive definitions like (5).

### 2.1 Building the Syntactic Algebra

(6) **The Syntactic Algebra**

Let  $\langle A, \text{Concat}, \text{Not}, \text{And} \rangle$  be the algebra such that:

- a. Concat, Not, And are the syntactic operations defined previously.
- b.  $A$  is the smallest set such that
  - (i) If  $\Phi$  is a predicate letter, then  $\Phi \in A$
  - (ii) If  $\alpha$  is an individual constant, then  $\alpha \in A$
  - (iii)  $A$  is closed under Concat, Not, And

(7) **Remarks**

- a.  $\text{WFF}_{\text{FOL-NoQ}} \subseteq A$ 
  - Anything in  $\text{WFF}_{\text{FOL-NoQ}}$  can be created by iterated application of Concat, Not, And to the predicate letters and individual constants.
- b.  $\text{WFF}_{\text{FOL-NoQ}} \neq A$ 
  - $A$  includes such 'syntactic garbage' as  $(P \ \& \ a)$ ,  $\sim x$ ,  $(ab)$ .
- c. Not every string in the vocabulary of FOL-NoQ is in  $A$  (*e.g.*,  $P_\sim$ ,  $\&(aB)$ )

(8) a. Question:

Since  $\text{WFF}_{\text{FOL-NoQ}} \subseteq A$ , why not equate FOL-NoQ with the algebra  $\langle A, \text{Concat}, \text{Not}, \text{And} \rangle$ , like we did for PL?

b. Answer:

There's not enough 'information' in  $\langle A, \text{Concat}, \text{Not}, \text{And} \rangle$  alone to construct/define the set  $\text{WFF}_{\text{FOL-NoQ}}$

- But, if we *paired* this algebra with some other formal devices that could be used to construct/define  $\text{WFF}_{\text{FOL-NoQ}}$  ...
- ...Then we could equate the language FOL-NoQ with that system!

*This will probably become a bit clearer to you once you actually see how we do this...*

## 2.2 Introducing the Syntactic Category Labels (The Types)

(9) **Background: Syntactic Categories and Syntactic Category Labels**

a. Syntactic Categories

We can view the 'syntactic categories' of a given language sets of strings in the vocabulary of that language.

*Illustration:*

- We could say that the category 'NP' in English is the set of strings { dog, student of history, ugly man with a telescope, big funny book about bees, ... }
- We could say that the category 'VP' in English is the set of strings { runs, ate a pickle, laughs in the face of danger, always sings horribly, ... }

b. Syntactic Category Labels

If syntactic categories are sets of strings, we could view the 'category labels' as being an index set, used to index the family of categories. (See Handout 1).

*Illustration:*

- Category Labels: {NP, N, VP, V, PP, P, DP, D, S, AP, A, .... }
- Syntactic Categories of English:
  - $C_{\text{NP}}, C_{\text{N}}, C_{\text{VP}}, C_{\text{V}}, C_{\text{PP}}, C_{\text{P}}, C_{\text{DP}}, C_{\text{D}}, C_{\text{S}}, C_{\text{AP}}, C_{\text{A}}, \dots$
  - $\{ C_{\delta} \}_{\delta \in \{\text{NP, N, VP, V, PP, P, DP, D, S, AP, A, ....}\}}$

We're now going to define a set of syntactic category labels and syntactic categories for FOL...  
These will later be used to define a set of syntactic rules that can be used to 'extract' WFF<sub>FOL-NoQ</sub> from the set A....

*The category labels will be very familiar to the semanticists...*

(10) **The Types**

The set T of types is the smallest set such that:

- a.  $e \in T$ 
  - o *This is the category label for expressions denoting 'entities'; i.e., the terms*
- b.  $t \in T$ 
  - o *This is the category label for things denoting 'truth-values'; i.e., the WFFs.*
- c. If  $\sigma \in T$  and  $\tau \in T$ , then  $\langle\sigma, \tau\rangle \in T$ 
  - o  *$\langle\sigma, \tau\rangle$  will be the category label for things denoting functions from  $\sigma$  to  $\tau$*
  - o *Thus,  $\langle e, t \rangle$  is the category label for functions from entities to truth-values (i.e., the unary predicates)...*

**2.3 Defining the Basic Expressions of FOL-NoQ**

(11) **Basic Expressions of a Category**

If  $\delta$  is a category label, then  $X_\delta$  are all the basic (primitive / lexical) expressions of category  $\delta$ .

*Illustration for English:*

- $X_N =$  *The set of nouns in the English lexicon.*
- $X_{NP} =$   $\{ \text{one} \}$ <sup>2</sup>
- $X_V =$  *The set of verbs in the English lexicon.*
- $X_{VP} =$   $\{ \text{so} \}$ <sup>3</sup>
- $X_P =$   $\{ \text{in, on, under, from, with ...} \}$
- $X_{PP} =$   $\{ \text{there, then, ...} \}$
- $X_D =$   $\{ \text{a, the, every, most, ...} \}$
- $X_{DP} =$   $\{ \text{he, she, it, they, ...} \}$
- $X_S =$   $\emptyset$

*We can use these sets  $X_\delta$  in conjunction with the syntactic rules (to be defined below) to generate the full set of categories  $C_\delta$  for the language.*

<sup>2</sup> This would be 'anaphoric *one*' in English, which some analyze as an NP pronouns (e.g. 'a student of chemistry from France, and **one** from Germany').

<sup>3</sup> This would be the VP-proform 'so', that appears in sentences like 'Mary went to the park, and Frank did **so** too'.

### (12) The Basic Expressions of FOL-NoQ

- a.  $X_e =$  The set of individual constants, CONS
- b.  $X_{\langle e, \dots, t \rangle} =$  The set of  $n$ -ary predicate letters, where  $n =$  the number of times  $e$  appears in the category label  $X_{\langle e, \dots, t \rangle}$   
*Illustration:*  $X_{\langle et \rangle} =$  The set of unary predicate letters.  
 $X_{\langle e \langle et \rangle} =$  The set of binary predicate letters.  
 $X_{\langle e \langle e \langle et \rangle \rangle} =$  The set of ternary predicate letters, etc.
- c. For all other types  $\tau \in T$ ,  $X_\tau = \emptyset$ .

Note: This states that our ‘lexicon’ for FOL-NoQ contains (i) individual constants, and (ii)  $n$ -ary predicate letters for every  $n \in \mathbb{N}$ , **but nothing else.** (i.e., the non-logical constants)

### 2.4 Introducing the Syntactic Rules of FOL-NoQ

Another important use of the syntactic category labels is in stating syntactic rules. Recall the syntactic rules that we had constructed for PL and FOL in the last handout.

### (13) Abstract Representation of Syntactic Rules for FOL (To Be Revised)

- a.  $\langle \text{Not}, \langle \text{WFF}, \text{WFF} \rangle, \text{WFF} \rangle$   
‘The result of applying Not to a member of WFF is a WFF’
- b.  $\langle \text{And}, \langle \text{WFF}, \text{WFF} \rangle, \text{WFF} \rangle$   
‘The result of applying And to a pair consisting of a WFF and a WFF is a WFF’

Recall that the WFFs of FOL-NoQ are going to be of category  $t$ . We could then rewrite the key rules in (13a,b) as follows:

### (14) Syntactic Rules for FOL-NoQ (To Be Revised)

- a.  $\langle \text{Not}, \langle t \rangle, t \rangle$   
‘The result of applying Not to a member of category  $t$  is also a member of category  $t$ ’.
- b.  $\langle \text{And}, \langle t, t \rangle, t \rangle$   
‘The result of applying And to a pair consisting of a member of category  $t$  and a member of category  $t$  is itself also a member of category  $t$ ’.

Finally, in addition to the rules in (14a,b), let's also add every rule of the general form in (15c), where  $\sigma, \tau \in T$ .

(15) **Syntactic Rules for FOL-NoQ (Final Version)**

- a.  $\langle \text{Not}, \langle t \rangle, t \rangle$
- b.  $\langle \text{And}, \langle t, t \rangle, t \rangle$
- c.  $\langle \text{Concat} \langle \langle \sigma, \tau \rangle, \sigma \rangle, \tau \rangle$   
‘The result of concatenating a member of category  $\langle \sigma, \tau \rangle$  with a member of category  $\sigma$  will be a member of category  $\tau$ .’

**IMPORTANT NOTE:**

(15c) is a kind of ‘meta-rule’, a short hand for representing *an infinite set* of rules. It encompasses all the following (concrete) rules:

- $\langle \text{Concat} \langle \langle e, t \rangle, e \rangle, t \rangle$   
‘The result of applying ‘Concat’ to a member of category  $\langle e, t \rangle$  and a member of category  $e$  will be a member of category  $t$ .’

*Illustration:* If  $P$  is of category  $\langle et \rangle$ , and  $a$  is of category  $e$ , then  $(Pa)$  is of category  $t$

- $\langle \text{Concat} \langle \langle e, \langle e, t \rangle \rangle, e \rangle, \langle e, t \rangle \rangle$   
‘The result of applying ‘Concat’ to a member of category  $\langle e, \langle e, t \rangle \rangle$  and a member of category  $e$  will be a member of category  $\langle e, t \rangle$ .’

*Illustration:* If  $R$  is of category  $\langle e, \langle e, t \rangle \rangle$ , and  $a$  is of category  $e$ , then  $(Ra)$  is of category  $\langle et \rangle$ .  
Consequently, if  $b$  is of category  $e$ , then  $((Ra)b)$  is of category  $t$

## 2.5 Using the Rules and Basic Expressions to Generate the Syntactic Categories

With our basic expressions in (12) and our syntactic rules in (15), we can define the syntactic category  $C_\delta$  for every category label  $\delta$ .

(16) **Definition of the Syntactic Categories  $C_\delta$**

For any category label  $\tau \in T$ ,  $C_\tau$  is the smallest set such that both the following hold:

- a.  $X_\tau \subseteq C_\tau$
- b. If (i)  $\langle f, \langle \sigma_1, \dots, \sigma_n \rangle, \tau \rangle$  is a syntactic rule, and (ii)  $\varphi_1, \dots, \varphi_n$  are such that each  $\varphi_i \in C_{\sigma_i}$ , then  $f(\varphi_1 \dots \varphi_n) \in C_\tau$

*In plain English, the (full) category  $C_t$  will contain all the basic (lexical) expressions of that category, as well as all the complex expressions of that category that you can create from the syntactic rules!*

- Note, then, that the definition in (16) is simply a formal, mathematically precise statement of how we've been informally reading rules like (15).

### (17) Illustration of the Definitions in (16)

- If  $P$  is a unary predicate letter, then  $P \in X_{\langle e \rangle}$  (12b), and so  $P \in C_{\langle e \rangle}$  (16a).
- If  $Q$  is a binary predicate letter, then  $Q \in X_{\langle e, \langle e, t \rangle \rangle}$ , and so  $Q \in C_{\langle e, \langle e, t \rangle \rangle}$
- If  $a, b$  are individual constants, then  $a, b \in X_e$  (12a), and so  $a, b \in C_e$  (16a).
- If  $P$  is a unary predicate letter, and  $a$  is an individual constant, then  $(Pa) \in C_t$ 
  - $P \in C_{\langle e \rangle}$  (17a)
  - $a \in C_e$  (17c)
  - $\langle \text{Concat} \langle \langle e, t \rangle, e \rangle, t \rangle$  is a syntactic rule (15c)
  - $\text{Concat}(P, a) \in C_t$  (16b)
  - $(Pa) \in C_t$
- If  $Q$  is a binary predicate letter, and  $a, b$  are individual constants, then  $((Qa)b) \in C_t$ 
  - $Q \in C_{\langle e, \langle e, t \rangle \rangle}$  (17b)
  - $a, b \in C_e$  (17c)
  - $\langle \text{Concat} \langle \langle e, \langle e, t \rangle \rangle, e \rangle, \langle e, t \rangle \rangle$  is a rule (15c)
  - $\text{Concat}(Q, a) \in C_{\langle e \rangle}$  (16b)
  - $(Qa) \in C_{\langle e \rangle}$
  - $\langle \text{Concat} \langle \langle e, t \rangle, e \rangle, t \rangle$  is a syntactic rule (15c)
  - $\text{Concat}((Qa), b) \in C_t$  (16b)
  - $((Qa)b) \in C_t$
- If  $P$  is a unary predicate letter, and  $a$  is an individual constant, then  $\sim(Pa) \in C_t$ 
  - $(Pa) \in C_t$  (17d)
  - $\langle \text{Not}, \langle t \rangle, t \rangle$  is a syntactic rule (15a)
  - $\text{Not}((Pa)) \in C_t$  (16b)
  - $\sim(Pa) \in C_t$
- If  $P$  is a unary predicate letter,  $Q$  is a binary predicate letter, and  $a, b$  are individual constants, then  $((Qa)b) \& \sim(Pa) \in C_t$ 
  - $\sim(Pa) \in C_t$  (17f)
  - $((Qa)b) \in C_t$  (17e)
  - $\langle \text{And}, \langle t, t \rangle, t \rangle$  is a syntactic rule (15b)
  - $\text{And}(((Qa)b), \sim(Pa)) \in C_t$  (16b)
  - $((Qa)b) \& \sim(Pa) \in C_t$

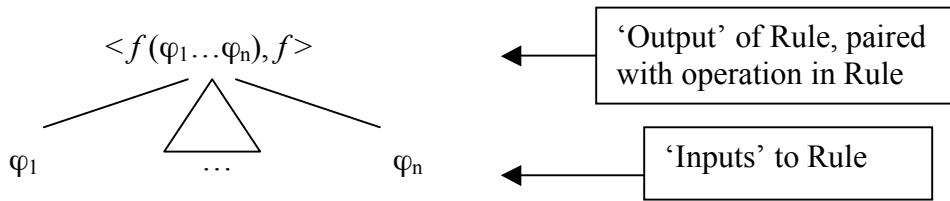
### (18) Introducing (Montagovian) Analysis Trees

- A handy way of representing calculations like those in (17a-g) is through the use of trees of form in (18a).
- These trees represent how a particular syntactic rules works to create an expression of a particular category.

#### a. General Form of An Analysis Tree

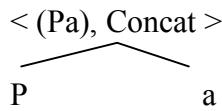
Suppose the rule  $\langle f, \langle \sigma_1, \dots, \sigma_n \rangle, \tau \rangle$  with (16) entails that  $f(\varphi_1 \dots \varphi_n) \in C_\tau$

This can be represented via a tree of the following form:

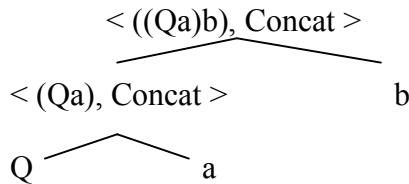


#### b. Illustrations

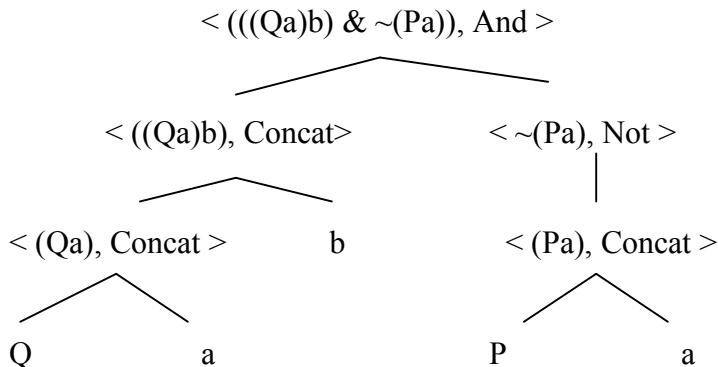
(i)



(ii)



(iii)



**Note:** Although these are superficially similar to PS trees, they are importantly different, in that the non-terminal nodes do not list the category of the resulting expression, but rather the syntactic operation used to obtain it...

## 2.6 Putting It All Together: Defining $\text{WFF}_{\text{FOL-NoQ}}$ in Terms of these Ingredients

### (19) The Crucial Observation: $C_t = \text{WFF}_{\text{FOL-NoQ}}$

That is, we can take our syntactic algebra  $\langle A, \text{Concat}, \text{Not}, \text{And} \rangle$  and obtain that subset of  $A$  equal to  $\text{WFF}_{\text{FOL-NoQ}}$  by adding:

- a. A set of category labels  $T$  (10)
- b. A set of ‘basic expressions’  $X_\tau$ , for each  $\tau \in T$  (12)
- c. A set of syntactic rules (15)

Given the definition in (16) for creating  $C_\tau$  from these ingredients (for every  $\tau \in T$ ), we are able to obtain the set  $C_t = \text{WFF}_{\text{FOL-NoQ}}$

Thus, if we add the ingredients in (19a,b,c) to our syntactic algebra  $\langle A, \text{Concat}, \text{Not}, \text{And} \rangle$ , we will have ‘enough information’ to obtain the set  $\text{WFF}_{\text{FOL-NoQ}}\dots$

...Thus, given the comment in (8b), we *could* abstractly characterize FOL-NoQ as in (20)

### (20) Algebraic Characterization of FOL-NoQ

The language FOL-NoQ is the structure  $\langle A, \text{Concat}, \text{Not}, \text{And}, X_\tau, S, t \rangle_{\tau \in T}$ , where:

- a.  $\langle A, \text{Concat}, \text{Not}, \text{And} \rangle$  is the algebra defined in (6)
- b.  $\langle \dots X_\tau \dots \rangle_{\tau \in T}$  is shorthand for the infinite family of basic expressions  $X_\tau$  defined in (12).
- c.  $S$  is the (infinite) set of syntactic rules defined in (15).
- d.  $t$  is the category label  $\tau$  such that  $C_\tau$  is the set of WFFs of FOL-NoQ (19).

Again, we’ve seen how the elements of the structure  $\langle A, \text{Concat}, \text{Not}, \text{And}, X_\tau, S, t \rangle_{\tau \in T}$  can be used to obtain  $\text{WFF}_{\text{FOL-NoQ}}$

- o For this reason, we are now justified in equating FOL-NoQ with this structure.
- o Note that  $\langle A, \text{Concat}, \text{Not}, \text{And}, X_\tau, S, t \rangle_{\tau \in T}$  is not actually itself an algebra. Rather, it contains an algebra as one of its proper parts.

### 3. Generalizing Our Treatment of FOL-NoQ to All Languages

In Montague Grammar, our method for defining FOL-NoQ by means of a ‘syntactic algebra’ is generalized to all languages, in (approximately) the following way.

#### (21) General Definition of a Language (To be Revised)

A language  $L$  is a structure  $\langle A, F_\gamma, X_\delta, S, \delta_0 \rangle_{\gamma \in \Gamma, \delta \in \Delta}$  such that:

- a.  $\langle A, F_\gamma \rangle_{\gamma \in \Gamma}$  is an algebra.
  - Note: the set  $\Gamma$  is an index set, used to index the (syntactic) operations in the (syntactic) algebra  $\langle A, F_\gamma \rangle_{\gamma \in \Gamma}$
- b.  $A$  is the smallest set such that:
  - (i) For all  $\delta \in \Delta$ ,  $X_\delta \subseteq A$ .
  - (ii)  $A$  is closed under the operations  $F_\gamma$  for all  $\gamma \in \Gamma$
  - Note: The set  $\Delta$  is the set of category labels, being used to index each of the set of ‘basic categories’ (lexical items)  $X_\delta$
  - Note: The conditions in (i) and (ii) are what give us that  $\langle A, F_\gamma \rangle_{\gamma \in \Gamma}$  is a ‘syntactic algebra’, where  $A$  contains all the well-formed structures of the language  $L$  (as well as a ton of junk).
- c.  $S$  is a set of sequences of the form  $\langle F_\gamma, \langle \delta_1, \dots, \delta_n \rangle, \delta \rangle$ , where  $\gamma \in \Gamma$ ,  $F_\gamma$  is an  $n$ -ary operation, and  $\delta_1, \dots, \delta_n, \delta \in \Delta$ 
  - Note: This condition states that a syntactic rule consists of:
    - (i) some  $n$ -ary operation  $F_\gamma$  from the algebra  $\langle A, F_\gamma \rangle_{\gamma \in \Gamma}$
    - (ii) a  $n$ -tuple of category labels from the set of labels  $\Delta$   
(the categories of the expressions ‘input’ to  $F_\gamma$ )
    - (iii) some category label  $\delta \in \Delta$   
(the category of the resulting output)
- d.  $\delta_0 \in \Delta$ 
  - Note:  $\delta_0$  will be the category label of the ‘sentences’ of  $L$ .
  - Note: Given that a language  $L$  is often characterized as a ‘set of sentences’, this element  $\delta_0$  will allow us to define such a set  $L$  given our structure above.

**Key Observation:**

Our ‘algebraic characterization’ of FOL-NoQ in (20) is a specific instance of the general kind of structure in (21).

With the general definition in (21) at our disposal, we can also offer the following general definition of what it is to be a ‘syntactic category’ of the language  $L$ .

**(22) General Definition of Syntactic Categories of a Language**

Let  $L$  be a language  $\langle A, F_\gamma, X_\delta, S, \delta_0 \rangle_{\gamma \in \Gamma, \delta \in \Delta}$ . Language  $L$  **generates the family CAT of syntactic categories iff:**

- a. CAT is a family of subsets of  $A$ , indexed by  $\Delta$ 
  - o Note: This means that the members of CAT are the sets  $C_\delta$ , for all  $\delta \in \Delta$
  - o Note: This all means that for all  $\delta \in \Delta$ ,  $C_\delta \subseteq A$ .
- b.  $X_\delta \subseteq C_\delta$ , for all  $\delta \in \Delta$ 
  - o Note: This means that for every category label  $\delta \in \Delta$ , the category  $C_\delta$  contains all the ‘basic expressions’ (lexical items) of that category.
- c. If the sequence  $\langle F_\gamma, \langle \delta_1, \dots, \delta_n \rangle, \delta \rangle \in S$ , and  $\varphi_1, \dots, \varphi_n$  are such that  $\varphi_i \in C_{\delta_i}$ , then  $F_\gamma(\varphi_1 \dots \varphi_n) \in C_\delta$ 
  - o Note: This is just a straightforward generalization of (16b)
  - o Note: This says that  $C_\delta$  will contain – not just the basic expressions lexical items  $X_\delta$  - but also all the expressions of category  $\delta$  you can generate via the syntactic rules.
- d. For each  $C_\delta \in \text{CAT}$ ,  $C_\delta$  is the smallest set satisfying conditions (22a)-(22c).

**(23) The ‘Meaningful Expressions’ of a Language  $L$**

Let  $L$  be a language  $\langle A, F_\gamma, X_\delta, S, \delta_0 \rangle_{\gamma \in \Gamma, \delta \in \Delta}$  and let  $L$  generate the family CAT of syntactic categories. The **meaningful expressions of  $L$  ( $ME_L$ )** is the union of all the categories in CAT (i.e.,  $\cup_{\delta \in \Delta} C_\delta$ )

#### 4. Applying Our Definition of a ‘Language’ to (a Fragment of) English

If (21) truly is a viable definition of what a ‘language’ is, then it should be possible to represent a *natural language*, like English, as such a system.

- In this section, we’ll show that this is possible for a subpart (fragment) of English, roughly corresponding in its expressive power to FOL-NoQ

##### (24) Some Comments

- a. In representing (a fragment of) English as a system like (21), we are basically giving a theory of English syntax.
  - I.e., the system responsible for constructing complex expressions of English from the primitive lexical items.
- b. One early and perennial criticism of Montague Grammar was its accompanying theory of syntax, stated in (21).
- c. Note, though, that the intellectual motivation behind Montague’s theory of syntax was *completely different* from that of generative linguists.
  - Montague wanted a theory broad enough to cover *anything that we’d reasonably call a ‘language’ (including artificial and non-human ones)*
  - Thus, it’s a *feature* – not a *bug* – that his system of syntax doesn’t (necessarily) capture generalizations regarding the structure of human languages.

##### (25) The Syntactic Category Labels

Let the set  $\Delta$  consist of the following syntactic category labels:

- a. NP (noun phrases; proper names for now)
- b. IV (intransitive verbs and derived verb phrases)
- c. TV (transitive verbs)
- d. S (sentences)

##### (26) The Basic Expressions (The Lexicon)

For each of the category labels above, the basic expressions of that category are:

- a.  $X_{NP} = \{ Barack, Michelle, Mitt \}$
- b.  $X_{IV} = \{ smokes \}$
- c.  $X_{TV} = \{ loves \}$
- d.  $X_S = \emptyset$

(27) **The Syntactic Operations**

a. Merge:

Binary operation that maps two strings ‘x’, ‘y’ to the string ‘x y’

*Illustration:* Merge( loves, Mitt ) = loves Mitt

b. And<sub>E</sub>

Binary operation that maps two strings ‘x’, ‘y’ to the string ‘x and y’

*Illustration:* And<sub>E</sub>( Mitt, Barack ) = Mitt and Barack

c. Not<sub>E</sub>

Unary operation that maps a string ‘x’ to the string ‘it is not the case that x’

*Illustration:* Not<sub>E</sub>( Barack smokes ) = It is not the case that Barack smokes

(28) **The Syntactic Rules**

Let the set S<sub>E</sub> consist of the following tuples:

a. < Merge, < TV, NP >, IV >

‘The result of merging a TV with an NP is a IV’

b. < Merge, < NP, IV >, S >

‘The result of merging an NP with an IV is an S’

c. < And<sub>E</sub> , <S , S> , S >

‘The result of applying And<sub>E</sub> to an S and an S is an S’

d. < Not<sub>E</sub>, <S>, S >

‘The result of applying Not<sub>E</sub> to an S is an S’.

(29) **The Definition of ‘Mini-English’**

The language ‘Mini-English’ is the structure < A, Merge, And<sub>E</sub>, Not<sub>E</sub>, X<sub>δ</sub> , S<sub>E</sub>, S >  $\delta \in \Delta$  such that:

a. A is the smallest set such that:

(i) For all  $\delta \in \Delta$ ,  $X_\delta \subseteq A$ .

(ii) A is closed under the operations Merge, And<sub>E</sub> and Not<sub>E</sub>

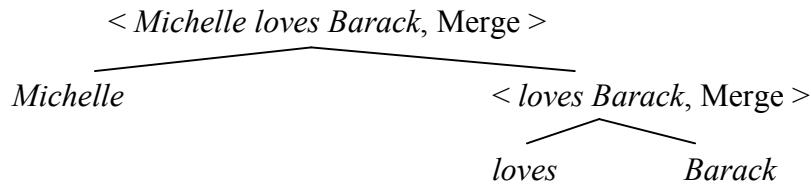
b. Merge, And<sub>E</sub>, Not<sub>E</sub>, X<sub>δ</sub> , S<sub>E</sub>, S, and  $\Delta$  are all as defined in (25)-(29)

(30) Some Remarks

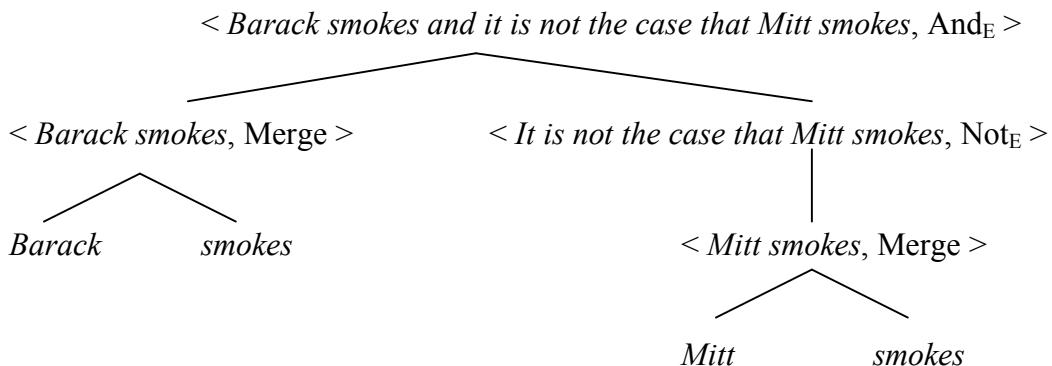
- a. The structure ‘Mini-English’ in (29) clearly satisfies our general definition of a ‘language’ in (21).
- b. The set A contains infinitely many well-formed sentences of English (*Mitt smokes*, *Barack loves Michelle* and *Michelle smokes*, etc.)
- c. The set A also contains a whole bunch of ‘syntactic garbage’ (*smokes loves*, *it is not the case that Mitt*, *Michelle Barack*, etc.)
- d. We can use the rules in  $S_E$  and the general definition in (22) to generate the syntactic categories  $C_\delta$  of English.
  - As with FOL-NoQ, we can use ‘analysis trees’ as a handy way of representing the calculations (31).
- e. Note that given rule (28a) the string *loves Michelle* is the same category (IV) as the intransitive verb *smokes*.
  - This is a recurring idea in Montague’s papers and other early MG works
  - That is, Montague and others don’t distinguish syntactically between ‘VPs’ and single intransitive verbs.
  - **I believe the main reason for this is that – from their perspective – this is more elegant than having an extra rule converting all IVs to VPs...**

(31) Some Sentences of Mini-English

- a. *Michelle loves Barack*.



- b. *Barack smokes and it is not the case that Mitt smokes*.



## Montague's General Theory of Semantics<sup>1</sup>

### 1. Revising Our Algebraic Perspective on 'Interpretation'

Last time, we motivated and illustrated the following general definition of what a 'language' is:

#### (1) The Montagovian Definition of 'Language' (To Be Slightly Revised)

A language  $L$  is a structure  $\langle A, F_\gamma, X_\delta, S, \delta_0 \rangle_{\gamma \in \Gamma, \delta \in \Delta}$  such that:

- a.  $\langle A, F_\gamma \rangle_{\gamma \in \Gamma}$  is an algebra.
- b.  $A$  is the smallest set such that:
  - (i) For all  $\delta \in \Delta$ ,  $X_\delta \subseteq A$ .
  - (ii)  $A$  is closed under the operations  $F_\gamma$  for all  $\gamma \in \Gamma$
- c.  $S$  is a set of sequences of the form  $\langle F_\gamma, \langle \delta_1, \dots, \delta_n \rangle, \delta \rangle$ , where  $\gamma \in \Gamma$ ,  $F_\gamma$  is an  $n$ -ary operation, and  $\delta_1, \dots, \delta_n, \delta \in \Delta$
- b.  $\delta_0 \in \Delta$

Given this (re)definition of what a 'language' is, we must now slightly revise our earlier definition of what an 'interpretation' of a language is.

#### (2) The Montagovian Definition of 'Interpretation' (To Be Slightly Revised)

Let  $L$  be a language  $\langle A, F_\gamma, X_\delta, S, \delta_0 \rangle_{\gamma \in \Gamma, \delta \in \Delta}$ . An *interpretation* for  $L$  is a structure  $\langle B, G_\gamma, f \rangle_{\gamma \in \Gamma}$  such that:

- a.  $\langle B, G_\gamma \rangle_{\gamma \in \Gamma}$  is an algebra with the same number of operations as  $\langle A, F_\gamma \rangle_{\gamma \in \Gamma}$

Note: This is basically already represented in the fact that the operations  $G_\gamma$  and  $F_\gamma$  are indexed by the same set  $\Gamma$

- b.  $G_\gamma$  is an operation of the same arity as  $F_\gamma$ , for all  $\gamma \in \Gamma$

Note: As we'd expect, each  $G_\gamma$  is the 'semantic operation' corresponding to the syntactic operation  $F_\gamma$

- c.  $f$  is a function from  $\bigcup_{\delta \in \Delta} X_\delta$  into  $B$ .

Note:  $f$  is a function from the basic expressions (lexical items) to some meanings. Thus,  $f$  represents the 'lexical semantics' of our language, and so is like the function 'I' in a model  $\langle D, I \rangle$ .

<sup>1</sup> These notes are based upon material in the following readings: Halvorsen & Ladusaw (1979), Dowty *et al.* (1981) Chapter 8, and Thomason (1974) Chapter 7 (Montague's "Universal Grammar").

(3) **The Definition of ‘Meaning Assignment’**

Let  $L$  be a language  $\langle A, F_\gamma, X_\delta, S, \delta_0 \rangle_{\gamma \in \Gamma, \delta \in \Delta}$ . Let  $\mathbf{B} = \langle B, G_\gamma, f \rangle_{\gamma \in \Gamma}$  be an interpretation for  $L$ .

The *meaning assignment* for  $L$  determined by  $\mathbf{B}$  is the unique homomorphism  $g$  from the syntactic algebra  $\langle A, F_\gamma \rangle_{\gamma \in \Gamma}$  to the semantic algebra  $\langle B, G_\gamma \rangle_{\gamma \in \Gamma}$  such that  $f \subseteq g$

(4) **Some Remarks**

- a. The fact that an interpretation  $\mathbf{B}$  does indeed determine such a (unique) homomorphism  $g$  is not obvious, but follows from conditions (2a,b).
- b. Under the definitions in (2) and (4), we still conceive of ‘meaning’ (i.e., semantics) as a homomorphism from a syntactic algebra to a semantic one.
- c. Under these new definitions, however, a meaning assignment  $g$  will *also* map all the ‘syntactic garbage’ in  $A$  to some kind of meanings in  $B$ .
  - **This consequence is definitely surprising. We’ll see in a bit, however, that it’s basically ‘harmless’.**
  - **Note, too, that it’s a necessary consequence of our general conception of semantics as ‘homomorphism between algebras’.**
    - Since the set of all and only the well-formed expressions of a language *don’t generally form an algebra*...
    - In the general case, the ‘syntactic algebra’ for a language will contain a bunch of ‘syntactic garbage’.
    - Consequently, a semantics for the language will also end up interpreting the ‘syntactic garbage’ in the algebra...

We’ll now illustrate the definitions in (2) and (3) by describing a general class of interpretations for FOL-NoQ, as well as some specific instances of this class...

## 2. Towards The Notion of a ‘Fregean Interpretation’

In his classic paper ‘Universal Grammar’, Montague introduces the term ‘Fregean Interpretation’ as a label for a specific *kind* of ‘interpretation’, as defined in (2).

- It’s a very complex definition, and we’ll begin to get a handle on it by introducing a core sub-part of it for our language of **FOL-NoQ**

### (5) The Language FOL-NoQ

The language FOL-NoQ is the structure  $\langle A, F_\gamma, X_\tau, S, t \rangle_{\gamma \in \{\text{Concat, Not, And}\}, \tau \in T}$  where:

a.  $\langle A, F_\gamma \rangle_{\gamma \in \{\text{Concat, Not, And}\}}$  is the algebra such that

- (i)  $F_{\text{Concat}} = \text{Concat}$  (from previous notes)
- $F_{\text{Not}} = \text{Not}$  (from previous notes)
- $F_{\text{And}} = \text{And}$  (from previous notes)

(ii)  $A$  is the smallest set such that:

1. It contains every individual constant and predicate letter.
2. It is closed under  $F_{\text{Concat}}, F_{\text{Not}}, F_{\text{And}}$

b. The basic categories  $X_\tau$  are such that:

(i)  $X_e =$  The set of individual constants, CONS

(ii)  $X_{\langle e, \dots, t \rangle} =$  The set of  $n$ -ary predicate letters, where  $n =$  the number of times  $e$  appears in the category label  $X_{\langle e, \dots, t \rangle}$

(iii) For all other types  $\tau \in T$ ,  $X_\tau = \emptyset$ .

c. The set  $S$  is the following (infinite) set of syntactic rules:

- (i)  $\langle F_{\text{Not}}, \langle t \rangle, t \rangle$
- (ii)  $\langle F_{\text{And}}, \langle t, t \rangle, t \rangle$
- (iii)  $\langle F_{\text{Concat}}, \langle \langle \sigma, \tau \rangle, \sigma \rangle, \tau \rangle$ , for all  $\sigma, \tau \in T$

*Illustrative Members of  $C_t$ :*

$$(((Qa)b) \& \sim(Rc)) \quad \sim(\sim(Ab) \& \sim((Sc)d)) \quad \sim(((Tg)b) \& \sim(Lg))$$

We’ll begin by defining a family of sets that will serve as the semantic values in a ‘Fregean interpretation’

(6) **The Set of ‘Possible Denotations’ (To be Revised)**

Let  $T$  be the set of types and  $E$  be some non-empty set (of entities). The set  $D_{\tau, E}$  of *denotations of type  $\tau$  based on  $E$*  is defined as follows:

- (i)  $D_{e,E} = E$
- (ii)  $D_{t,E} = \{0, 1\}$
- (iii) If  $\sigma, \tau \in T$ , then  $D_{<\sigma, \tau>, E} =$  the set of functions from  $D_{\sigma, E}$  to  $D_{\tau, E}$   
 $= D_{\tau, E}^{D_{\sigma, E}}$

*Illustration:* Let  $E = \{\text{Michelle, Barack, Mitt}\}$ .

$$D_{e,E} = \{\text{Michelle, Barack, Mitt}\}$$

$$D_{t,E} = \{0, 1\}$$

$D_{<et>, E} =$  All the functions from  $\{\text{Michelle, Barack, Mitt}\}$  to  $\{0, 1\}$   
All the characteristic functions of members of  $P(E)$ .

$D_{<e, <et>, E} =$  All the functions from  $\{\text{Michelle, Barack, Mitt}\}$  to  
some function from  $\{\text{Michelle, Barack, Mitt}\}$  to  
 $\{1, 0\}$   
= All the characteristic functions of members of  $E \times E$

The family of sets defined in (6) will be a key ingredient in the definition of a ‘Fregean Interpretation’ of FOL-NoQ...

The second key ingredient will be the semantic operations defined below...

(7) **The Semantic Operations (To Be Revised)**

- a. The Operation ‘ $G_{\text{Not}}$ ’  
 $G_{\text{Not}} = \{\langle 1, 0 \rangle, \langle 0, 1 \rangle\}$

- b. The Operation ‘ $G_{\text{And}}$ ’  
 $G_{\text{And}} = \{\langle \langle 1, 1 \rangle, 1 \rangle, \langle \langle 1, 0 \rangle, 0 \rangle, \langle \langle 0, 1 \rangle, 0 \rangle, \langle \langle 0, 0 \rangle, 0 \rangle\}$

- c. The Operation ‘ $G_{\text{Concat}}$ ’<sup>2</sup>  
If  $\alpha \in D_{<\sigma, \tau>, E}$  and  $\beta \in D_{\sigma, E}$ , then  $G_{\text{Concat}}(\alpha, \beta) = \alpha(\beta)$

With these ingredients, we can now make a first attempt at defining what a ‘Fregean Interpretation’ of FOL-NoQ is...

<sup>2</sup> Note that  $G_{\text{Concat}}$  is basically an operation of ‘function application’.

(8) **Fregean Interpretation of FOL-NoQ (To Be Substantially Revised)**

Let  $E$  be a set (of entities). A *Fregean Interpretation of FOL-NoQ based on  $E$*  is an interpretation of  $\text{FOL-NoQ} < B, G_\gamma, f \rangle_\gamma \in \{\text{Concat, Not, And}\}$  such that

$$(i) \quad B = \bigcup_{\tau \in T} D_{\tau, E}$$

Note: The set  $B$  of semantic values is the union of *all* the denotations of type  $\tau$  based on  $E$ .

$$(ii) \quad \text{The operations } G_{\text{Concat}}, G_{\text{Not}}, G_{\text{And}} \text{ are as defined in (7).}$$

(9) **An Immediate Problem for the Definition in (8)**

- According to our definition in (2), in order for  $< B, G_\gamma, f \rangle_\gamma \in \{\text{Concat, Not, And}\}$  to be an ‘interpretation’,  $< B, G_\gamma \rangle_\gamma \in \{\text{Concat, Not, And}\}$  must be an *algebra*.
- However, if  $B = \bigcup_{\tau \in T} D_{\tau, E}$ , then it isn’t closed under  $G_{\text{Concat}}, G_{\text{Not}}, G_{\text{And}}$ 
  - $G_{\text{Not}}, G_{\text{And}}$  are only defined for elements of  $B$  that are in  $D_{t, E}$
  - $G_{\text{Concat}}$  is only defined for pairs  $\alpha, \beta$  such that  $\alpha \in D_{\langle \sigma, \tau \rangle, E}$  and  $\beta \in D_{\sigma, E}$ ,

(10) **A Related Problem**

- If  $< B, G_\gamma, f \rangle_\gamma \in \{\text{Concat, Not, And}\}$  is to be an interpretation, then the meaning assignment based on it must be homomorphism from  $A$  in (5) to  $B = \bigcup_{\tau \in T} D_{\tau, E}$
- **But remember that  $A$  contains a whole bunch of ‘syntactic garbage’.**
- Thus, there has to be ‘meanings’ in  $B$  that this syntactic garbage gets mapped to
- **For example, since  $G_{\text{Not}}$ , and  $F_{\text{Not}}$  would ‘correspond’ in the homomorphism, we need to have it that for an individual constant ‘ $a$ ’ where  $f(a) = \alpha \in D_{e, E}$**

$$g(\sim a) =$$

$$g(F_{\text{Not}}(a)) =$$

$$G_{\text{Not}}(g(a)) =$$

$$G_{\text{Not}}(f(a)) =$$

$$G_{\text{Not}}(\alpha) = ??? = \text{some kind of ‘semantic garbage’ corresponding to the ‘syntactic garbage’ } \sim a$$

(11) **Incorporating ‘Semantic Garbage’ Into Our System**

What is ‘semantic garbage’, however? And how do we incorporate it into our system?

- Montague is conspicuously silent on the matter in UG and PTQ. To my knowledge, there’s no indication in his extant writings on how he thought the issue in (9)/(10) could/should be concretely handled (*probably it was too trivial to discuss...*)
- Halvorsen & Ladusaw (1979) have some more focused discussion of the matter, but they also leave the nature of the ‘semantic garbage’ entirely open (and cryptic).
- For better or worse, the following is my own proposal: **we are going to slightly redefine  $D_e$  so that it contains a special element ‘garbage’**
  - We are then going to redefine the operations  $G_\gamma$  in light of this...

(12) **The Set of ‘Possible Denotations’ (To be Revised)**

Let  $T$  be the set of types and  $E$  be some non-empty set (of entities). The set  $D_{\tau, E}$  of *denotations of type  $\tau$  based on  $E$*  is defined as follows:

- (i)  $D_{e,E} = E \cup \{ \text{garbage} \}$
- (ii)  $D_{t,E} = \{ 0, 1 \}$
- (iii) If  $\sigma, \tau \in T$ , then  $D_{<\sigma, \tau>, E} =$  the set of functions from  $D_{\sigma, E}$  to  $D_{\tau, E}$   
 $= D_{\tau, E}^{D_{\sigma, E}}$

(13) **The Semantic Operations**

a. The Operation ‘ $G_{\text{Not}}$ ’

- (i) If  $\alpha \notin D_{t,E}$ , then  $G_{\text{Not}}(\alpha) = \text{garbage}$
- (ii) If  $\alpha \in D_{t,E}$ , then  $G_{\text{Not}}(\alpha) = \begin{cases} 1, & \text{if } \alpha = 0 \\ 0, & \text{otherwise} \end{cases}$

b. The Operation ‘ $G_{\text{And}}$ ’

- (i) If  $\alpha \notin D_{t,E}$  or  $\beta \notin D_{t,E}$ , then  $G_{\text{And}}(\alpha, \beta) = \text{garbage}$
- (ii) If  $\alpha, \beta \in D_{t,E}$ , then  $G_{\text{And}}(\alpha, \beta) = \begin{cases} 1, & \text{if } \alpha = \beta = 1 \\ 0, & \text{otherwise} \end{cases}$

c. The Operation ‘ $G_{\text{Concat}}$ ’

- (i) If  $\alpha = \text{garbage}$  or  $\beta = \text{garbage}$ , then  $G_{\text{Concat}}(\alpha, \beta) = \text{garbage}$
- (ii) If not  $\alpha \in D_{<\sigma, \tau>, E}$  and  $\beta \in D_{\sigma, E}$ , then  $G_{\text{Concat}}(\alpha, \beta) = \text{garbage}$
- (iii) Otherwise, if  $\alpha \in D_{<\sigma, \tau>, E}$  and  $\beta \in D_{\sigma, E}$ , then  $G_{\text{Concat}}(\alpha, \beta) = \alpha(\beta)$

(14) **Illustration of These Definitions**

Let  $f$  be the  $\langle e, t \rangle$  function  $\{ \langle \text{Barack}, 0 \rangle, \langle \text{Michelle}, 1 \rangle \}$ <sup>3</sup>

- |    |  |    |   |
|----|--|----|---|
| a. | $G_{\text{Not}}(1) = 0$  | b. | $G_{\text{Not}}(\text{Barack}) = \text{garbage}$        |
| c. | $G_{\text{And}}(1, 1) = 1$   | d. | $G_{\text{And}}(1, f) = \text{garbage}$                 |
| e. | $G_{\text{Concat}}(f, \text{Barack}) = f(\text{Barack}) = 0$         | f. | $G_{\text{Concat}}(f, \text{garbage}) = \text{garbage}$ |
| g. | $G_{\text{Concat}}(\text{Michelle}, \text{Barack}) = \text{garbage}$ | h. | $G_{\text{Concat}}(f, f) = \text{garbage}$              |

*With the formalization of ‘semantic garbage’ in (12) and (13) at hand, we can now properly define a ‘fregean interpretation’ for FOL-NoQ...*

(15) **Fregean Interpretation of FOL-NoQ (To Be Substantially Revised)**

Let  $E$  be a non-empty set (of entities). A *Fregean Interpretation of FOL-NoQ based on E* is an interpretation of FOL-NoQ  $\langle B, G_\gamma, f \rangle_{\gamma \in \{\text{Concat, Not, And}\}}$  such that

a.  $B = \bigcup_{\tau \in T} D_{\tau, E}$

Note: Again, the set  $B$  of semantic values is the union of *all* the denotations of type  $\tau$  *based on E*.

Note: This definition now makes use of the revised definition in (12), which adds the special element **garbage** to  $D_{e, E}$

- b. The operations  $G_{\text{Concat}}$ ,  $G_{\text{Not}}$ ,  $G_{\text{And}}$  are as defined in (13).

Note: Under the new definitions in (13),  $B$  is closed under  $G_{\text{Concat}}$ ,  $G_{\text{Not}}$ ,  $G_{\text{And}}$

- c. The function  $f$  is such that

- (i) For all  $\tau \in T$ , if  $\alpha \in X_\tau$  then  $f(\alpha) \in D_{\tau, E}$
- (ii) For all  $\alpha \in X_e$ ,  $f(\alpha) \neq \text{garbage}$

Note: The conditions in (15c) entail that:

- Every individual constant will be mapped to some entity in  $D_e$
- No individual constant will be mapped to (interpreted as) **garbage** (in  $D_e$ )
- A n-ary predicate letter of type  $\langle e, \dots, t \rangle$  will be mapped to a function in  $D_{\langle e, \dots, t \rangle, E}$ 
  - Thus, an n-ary predicate letter is mapped to the curried characteristic function of an n-ary relation! (Just as it should be).

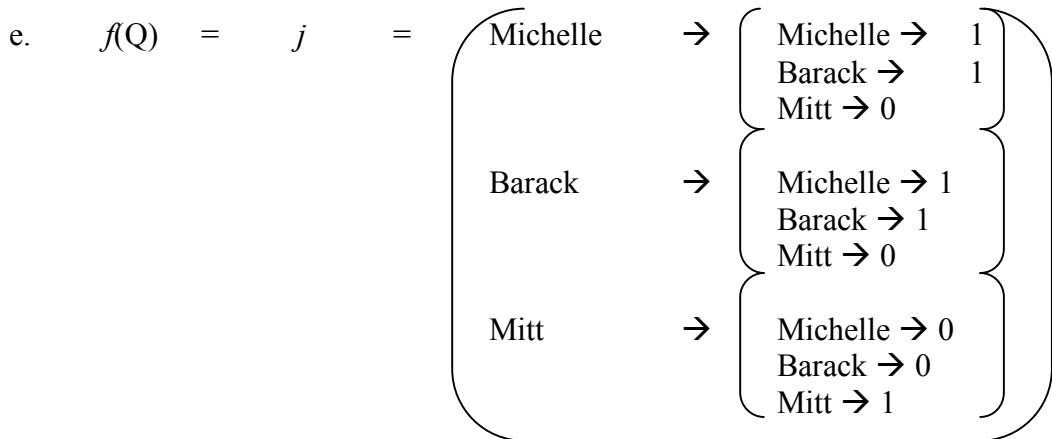
---

<sup>3</sup> Technically speaking  $f$  must also now be defined for **garbage**. We can simply assume that  $f(\text{garbage}) = 0$  for all  $\langle e, t \rangle$  functions  $f$ . Similar such assumptions can be made for functions of the other types. From this point on, we’ll simply ignore **garbage** in our specifications of the denotations of the basic lexical items.

(16) A Concrete Instance of a Fregean Interpretation of FOL-NoQ

Let the set  $S = \{ \text{Michelle}, \text{Barack}, \text{Mitt} \}$ . Let  $\mathbf{B} = \langle B, G_\gamma, f \rangle_{\gamma \in \{\text{Concat}, \text{Not}, \text{And}\}}$  be the Fregean interpretation based on  $S$ , such that  $f$  contains the following mappings:

- a.  $f(a) = \text{Michelle}$
- b.  $f(b) = \text{Barack}$
- c.  $f(c) = \text{Mitt}$
- d.  $f(P) = h = \{ \langle \text{Michelle}, 0 \rangle, \langle \text{Barack}, 1 \rangle, \langle \text{Mitt}, 0 \rangle \}$



Note: We've basically interpreted 'P' as the property 'is president', and we've interpreted 'Q' as the relation 'x is loved by y'.

We can now use the 'meaning assignment  $g$  determined by  $\mathbf{B}$ ' to interpret formulae in our FOL-NoQ language!...

(17) Using the Fregean Interpretation in (16) to Interpret Sentences of FOL-NoQ

Let  $g$  be the meaning assignment for FOL-NoQ that is determined by  $\mathbf{B}$  in (16).

- (i)  $g(\sim(Pa)) =$  (by definition of FOL-NoQ)
- (ii)  $g(F_{\text{Not}}(F_{\text{Concat}}(P,a))) =$  (by homomorphism property of  $g$ )
- (iii)  $G_{\text{Not}}(g(F_{\text{Concat}}(P,a))) =$  (by homomorphism property of  $g$ )
- (iv)  $G_{\text{Not}}(G_{\text{Concat}}(g(P), g(a))) =$  (by definition of  $g$ )<sup>4</sup>
- (v)  $G_{\text{Not}}(G_{\text{Concat}}(f(P), f(a))) =$  (by definition of  $f$  in (16))
- (vi)  $G_{\text{Not}}(G_{\text{Concat}}(h, \text{Michelle})) =$  (by definition of  $G_{\text{Concat}}$ )
- (vii)  $G_{\text{Not}}(h(\text{Michelle})) =$  (by definition of  $h$  in (16d))
- (viii)  $G_{\text{Not}}(0) =$  (by definition of  $G_{\text{Not}}$ )
- (ix)  $1$

<sup>4</sup> Note that by the definition in (3),  $f \subseteq g$ , and so if  $\alpha \in X$ , then  $g(\alpha) = f(\alpha)$ .

(18) **Using the Fregean Interpretation in (16) to Interpret Sentences of FOL-NoQ**  
Let  $g$  be the meaning assignment for FOL-NoQ that is determined by  $\mathbf{B}$  in (16).

- (i)  $g( (((Qa)b) \& (Pb))) =$  (by definition of FOL-NoQ)
- (ii)  $g( F_{\text{And}}( F_{\text{Concat}}(F_{\text{Concat}}(Q,a),b), F_{\text{Concat}}(P,b) ) ) =$  (by homomorphism property of  $g$ )
- (iii)  $G_{\text{And}}( g( F_{\text{Concat}}(F_{\text{Concat}}(Q,a),b) ), g( F_{\text{Concat}}(P,b) ) ) =$  (by homomorphism property of  $g$ )
- (iv)  $G_{\text{And}}( G_{\text{Concat}}( g(F_{\text{Concat}}(Q,a)), g(b) ), G_{\text{Concat}}( g(P), g(b) ) ) =$   
(by homomorphism property of  $g$ )
- (v)  $G_{\text{And}}( G_{\text{Concat}}(G_{\text{Concat}}(g(Q), g(a)), g(b)), G_{\text{Concat}}( g(P), g(b) ) ) =$  (by definition of  $g$ )
- (vi)  $G_{\text{And}}( G_{\text{Concat}}(G_{\text{Concat}}(f(Q), f(a)), f(b)), G_{\text{Concat}}( f(P), f(b) ) ) =$  (by definition of  $f$ )
- (vii)  $G_{\text{And}}( G_{\text{Concat}}(G_{\text{Concat}}(j, \text{Michelle}), \text{Barack}), G_{\text{Concat}}( h, \text{Barack} ) ) =$   
(by definition of  $G_{\text{Concat}}$ )
- (viii)  $G_{\text{And}}( G_{\text{Concat}}(G_{\text{Concat}}(j, \text{Michelle}), \text{Barack}), h(\text{Barack}) ) =$  (by definition of  $h$ )
- (ix)  $G_{\text{And}}( G_{\text{Concat}}(G_{\text{Concat}}(j, \text{Michelle}), \text{Barack}), 1 ) =$  (by definition of  $G_{\text{Concat}}$ )
- (x)  $G_{\text{And}}( G_{\text{Concat}}(j(\text{Michelle}), \text{Barack}), 1 ) =$  (by definition of  $j$ )
- (xi)  $G_{\text{And}}( G_{\text{Concat}}( \{<\text{Michelle}, 1>, <\text{Barack}, 1>, <\text{Mitt}, 0>\}, \text{Barack} ), 1 ) =$   
(by definition of  $G_{\text{Concat}}$ )
- (xii)  $G_{\text{And}}( 1, 1 ) = 1$  (by definition of  $G_{\text{And}}$ )

(19) **Using the Fregean Interpretation in (16) Interpret Syntactic Garbage**

- (i)  $g( (P \sim a) ) =$  (by definition of FOL-NoQ)
- (ii)  $g( F_{\text{Concat}}( P, F_{\text{Not}}(a) ) ) =$  (by homomorphism property of  $g$ )
- (iii)  $G_{\text{Concat}}( g(P), G_{\text{Not}}( g(a) ) ) =$  (by definition of  $g$ )
- (iv)  $G_{\text{Concat}}( f(P), G_{\text{Not}}( f(a) ) ) =$  (by definition of  $f$ )
- (v)  $G_{\text{Concat}}( h, G_{\text{Not}}(\text{Michelle}) ) =$  (by definition of  $G_{\text{Not}}$  in (13a))
- (vi)  $G_{\text{Concat}}( h, \mathbf{garbage} ) =$  (by definition of  $G_{\text{Concat}}$  in (13c))
- (vii)  $\mathbf{garbage}$

### 3. Fregean Interpretations and Models of FOL-NoQ

#### (20) Taking Stock of What We've Done So Far

- In the last set of notes, we developed the general definition of ‘language’ in (1), according to which any language is built upon a ‘syntactic algebra’  $\langle A, F_\gamma \rangle_{\gamma \in \Gamma}$
- We developed a notion of ‘meaning’, according to which a semantics for a given language is a homomorphism from the syntactic algebra  $\langle A, F_\gamma \rangle_{\gamma \in \Gamma}$  to a semantic algebra  $\langle B, G_\gamma \rangle_{\gamma \in \Gamma}$
- We developed a notion of ‘Fregean interpretation’, a special *kind* of interpretation for our FOL-NoQ language.
- We saw how one specific such ‘Fregean interpretation’ can be used to map sentences of FOL-NoQ to truth-values (just like a classical model!)

*There's an important relation between our concept of 'Fregean Interpretation' in (15) and our concept of a 'model' for FOL, repeated below:*

#### (21) Definition of a ‘Model’ for First Order Logic

A model  $\mathcal{M}$  is a pair  $\langle D, I \rangle$  consisting of:

- a. A *non-empty* set  $D$ , called the ‘domain of  $\mathcal{M}$ ’
- b. A function  $I$ , whose domain is the individual constants and predicate letters, and whose range satisfies the following conditions:
  - (i) If  $\alpha$  is an individual constant, then  $I(\alpha) \in D$
  - (ii) If  $\Phi$  is an  $n$ -ary predicate letter, then  $I(\Phi)$  is **the curried characteristic function of an  $n$ -ary relation  $R \subseteq D^n$**

#### (22) The Correspondence Between (Fregean) Interpretations and Models, Part 1

Let  $\mathbf{B} = \langle B, G_\gamma, f \rangle_{\gamma \in \{\text{Concat}, \text{Not}, \text{And}\}}$  be a Fregean interpretation (for FOL-NoQ); let  $g$  be meaning assignment determined by  $\mathbf{B}$ . Let  $\mathcal{M} = \langle D, I \rangle$  be such that  $D = D_{e,E}$  and  $I = f$ .

- a. Claim:  $\langle D, I \rangle$  is a model (for FOL-NoQ)
  - $D = D_{e,E}$  is by definition a non-empty set.
  - By definition in (15c),  $f$  is a function satisfying the condition in (21b).
- b. Claim: For any sentence  $\varphi$  (of FOL-NoQ),  $g(\varphi) = [[\varphi]]^M$   
(proof left as exercise for the student)

### (23) The Correspondence Between (Fregean) Interpretations and Models, Part 2

Let  $\mathcal{M} = \langle D, I \rangle$  be a model (for FOL-NoQ). Let  $\mathbf{B} = \langle B, G_\gamma, f \rangle_{\gamma \in \{\text{Concat, Not, And}\}}$  be such that  $B = \bigcup_{\tau \in T} D_{\tau, D}$  and  $f = I$ .

- a. Claim:  $\langle B, G_\gamma, f \rangle_{\gamma \in \{\text{Concat, Not, And}\}}$  is a Fregean interpretation (for FOL-NoQ)
  - By definition,  $B = \bigcup_{\tau \in T} D_{\tau, E}$ , for a (non-empty) set  $E = D$
  - By definition,  $G_{\text{Concat}}, G_{\text{Not}}, G_{\text{And}}$  are as defined in (13).
  - By definition (21b), function  $I$  satisfies the conditions in (15c); it is such that
    - (i) For all  $\tau \in T$ , if  $\alpha \in X_\tau$  then  $f(\alpha) \in D_{\tau, E}$
    - (ii) For all  $\alpha \in X_e, f(\alpha) \neq \text{garbage}$
- b. Claim: Let  $g$  be the meaning assignment determined by  $\mathbf{B}$ . For any sentence  $\varphi$  (of FOL-NoQ),  $g(\varphi) = [[\varphi]]^M$   
(proof left as exercise for the student)

### (24) Some Remarks

- a. Thus, for every Fregean interpretation  $\mathbf{B}$  based on  $E$ , there is a corresponding model  $\mathcal{M}$  whose domain is  $E$ , and which makes exactly the same sentences true.
- b. Similarly, for every model  $\mathcal{M} = \langle D, I \rangle$ , there is a Fregean interpretation  $\mathbf{B}$  based on  $D$ , and which makes exactly the same sentences true.
- c. Given this relationship between models and Fregean interpretations, we can freely shift between the two (*cf.* sets and their characteristic functions).
- d. Similarly, we can view Fregean interpretations and models as being *in essence* ‘the same thing’ (even though they are different set-theoretic objects)  
(*cf.* sets and their characteristic functions)
- e. **Thus, if we can provide a Fregean interpretation for a given language, we’ve also thereby provided that language with a model-theoretic semantics.**
- f. Finally, given this correspondence between models and *Fregean* interpretations, we can see that our general notion of ‘interpretation’ in (2) is truly a more general concept of ‘interpretation’ than is found in our model-theoretic semantics of FOL.

#### 4. Extending The Framework to Natural Languages

##### (25) Burning Question:

Can we provide a (Fregean) interpretation to (a fragment of) a natural language, like English?

*A second fundamental contribution of Montague's was the discovery that there are, in principle, two different ways of providing an interpretation for a language.*

##### (26) Method 1: Direct Interpretation of a (Natural) Language $L$

###### a. Step 1:

Provide an analysis of language  $L$  as a structure  $\langle A, F_\gamma, X_\delta, S, \delta_0 \rangle_{\gamma \in \Gamma, \delta \in \Delta}$  satisfying the core definition in (1).

###### b. Step 2: Specify a structure $\langle B, G_\gamma, f \rangle_{\gamma \in \Gamma}$ satisfying the core definition in (2) of an ‘interpretation’ for $\langle A, F_\gamma, X_\delta, S, \delta_0 \rangle_{\gamma \in \Gamma, \delta \in \Delta}$

- This is the method pursued by Montague in his “English as a Formal Language”

##### (27) Method 2: Indirect Interpretation (To be Revised)

###### a. Step 1:

Provide an analysis of language  $L$  as a structure  $\langle A, F_\gamma, X_\delta, S, \delta_0 \rangle_{\gamma \in \Gamma, \delta \in \Delta}$  satisfying the core definition in (1).

###### b. Step 2:

Take an (artificial) language  $L' = \langle A', F'_\gamma, X'_{\delta'}, S', \delta'_0 \rangle_{\gamma' \in \Gamma', \delta' \in \Delta'}$  such that *there is already a known interpretation  $B = \langle B, G_{\gamma'}, f \rangle_{\gamma' \in \Gamma'}$  for  $L'$ .*

###### c. Step 3: (To be Revised)

Give a homomorphism  $h$  from  $A$  (the structures of  $L$ ) to  $A'$  (the structures of  $L'$ )  
o *This homomorphism  $h$  is effectively a ‘translation’ from  $L$  into  $L'$ .*

- This is the method pursued by Montague in his most seminal papers, UG and PTQ

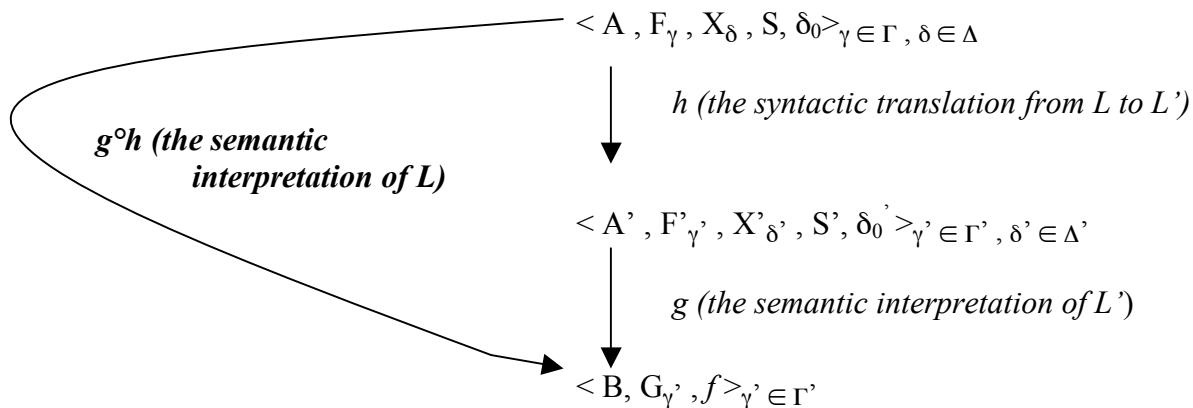
##### (28) Key Skeptical Question

With this ‘indirect interpretation’, all we really do is supply a ‘translation’  $h$  from one language  $L$  into another  $L'$ . *Have we really provided  $L$  with a semantics when we do this?*

(29) Answer to the Skeptical Question

- Recall that the composition of two homomorphisms is itself a homomorphism (Assignment 4)
- Therefore, if  $\mathbf{B}$  is the interpretation of  $L'$ , and  $g$  is the (homomorphic) meaning assignment determined by  $\mathbf{B}$ , and  $h$  is the (homomorphic) translation from  $L$  to  $L'$ , ***then  $g \circ h$  is a homomorphism from A (the expressions of L) to B (the meanings assigned to L').***
- Therefore, by providing an indirect interpretation for a language  $L$ , ***we've also provided a (model-theoretic) semantics for L***

Indirect Interpretation in a Picture (Oversimplified):



In other words, with ‘indirect interpretation’, we get our semantics for  $L$  ‘indirectly’, through the translation language  $L'$ . ***But we still do get a semantics for L.***

This notion of ‘indirect interpretation’ will become clearer once we work through a concrete example...

There are a few important properties of ‘indirect interpretation’ that we can go ahead and observe, however....

(30) Indirect Interpretation is ‘Eliminable’

- Note that if we ‘indirectly interpreted’ language  $L$  via language  $L'$ , then we have also thereby ‘directly interpreted’  $L$ , via the composition of the two homomorphisms  $g \circ h$
- Thus, indirect interpretation is never a necessary part of giving a semantics for a language  $L$ ; it’s ‘eliminable’:
  - Once we’ve done ‘indirect interpretation’ we could recast everything we’ve done as ‘direct interpretation’ via the composition  $g \circ h$

### (31) Some Choice Quotes

- From Halvorsen & Ladusaw (1979), p. 210:  
“An understanding of [the eliminability of indirect interpretation] is necessary to understand the use of...logics of PTQ and most other analyses within Montague grammar. As has been stated elsewhere, the [logical language] is an **expository device** and is in no way a necessary part of an analysis of any language offered within this theory. By using the easily interpreted [logical language] as a mediator, natural languages can be analyzed syntactically and then provided with a translation...from them to [the logical language] to induce their interpretation....**This method of analysis amounts to direct interpretation of natural language.**” (emphasis mine)
- From Dowty *et al.* (1981), p. 263:  
“Translating English into [a logical language] was therefore not essential to interpreting the English phrases we generated; **it was simply a convenient intermediate step in assigning them meanings. This step could have been eliminated had we chosen to describe the interpretation of English directly...** This point is important, because anyone who does not appreciate it may misunderstand the role of [logical languages] in applications of Montague’s descriptive framework to natural languages.” (emphasis mine)

The general point here is that, to give a semantics for (e.g.) English in Montague Grammar, you don’t *have* to translate English into any logical language...

- It’s just that doing so can be a very handy, elegant means of specifying an interpretation for the language.

*Again, this will all become clearer when we’ve seen some concrete instances of it...*

### (32) Why Do Indirect Interpretation?

If the logical language is well-designed and familiar to readers, then it can provide a more ‘perspicuous’ representation (statement / name) of the meanings that we wish to be assigned to the (natural) language expressions.

*Illustration (From 610):*

‘ $[\lambda x_e : x \text{ smokes}]$ ’ vs. ‘The function  $f$  from  $D_e$  to  $D_t$  such that for all  $x \in D_e$ ,  $f(x) = 1$  iff  $x$  smokes.’

- From Dowty *et al.* (1981), p. 264:  
“[The purpose of indirect interpretation] was to have a convenient, compact notation for giving a briefer statement of semantic rules than we were able to give in earlier chapters of this book, where semantic rules were formulated rather long-windedly in English... [The logical language] could provide us with names for meanings...”

(33) **Partisan Comment: Indirect Interpretation and Semantics in the 21<sup>st</sup> Century**

- In a certain sense, much of the semantic literature nowadays is rife with ‘indirect interpretation’, *but without an explicit, model-theoretic semantics for the translation language.*
- When in semantics paper we write things like  $[[\text{-er}]] = [\lambda P_{\langle dt \rangle} : \lambda Q_{\langle dt \rangle} : \dots]$ , we assume that whatever appears to the right of ‘=’ is understood to our readers (and ourselves).
  - *This isn’t always the case though... sometimes formulas and notations creep in that seem to make sense, but turn out not to upon deeper probing...*
  - In the original Montague Grammar framework, there is much lower risk of this happening, *since one must also provide an explicit model-theoretic semantics for their interpretation language.*

*So, this is the general gist of how ‘indirect interpretation’ in MG works...*

*Note, however, that the characterization in (27) has some gross oversimplifications...*

In the next set of notes, we’ll set the record straight by delving into Montague’s theory of translation...

### Problem Set on Languages and Interpretations

(1) **Basic Comprehension Questions on Our (Provisional) Definition of a ‘Language’**

Let  $L$  be a language  $\langle A, F_\gamma, X_\delta, S, \delta_0 \rangle_{\gamma \in \Gamma, \delta \in \Delta}$  as defined in (21) on the handout “An Algebraic Perspective on the Syntax of First Order Logic”.

- a. What are the syntactic rules of  $L$ ?
- b. What are the syntactic category labels of  $L$ ?
- c. What are the basic expressions (‘lexical items’) of  $L$ ? (Please represent as an indexed family of sets.)
- d. Which elements in  $L$  form an algebra together?
- f. What is the category label for the declarative sentences of  $L$ ?
- g. What are the syntactic operations of  $L$ ? (Please represent as an indexed family of sets).
- h. What is the difference between  $A$  and the meaningful expressions of  $L$ ? Can they ever be the same?
- i. Let  $CAT$  be the syntactic categories of  $L$ .
  - (i) Which element are the members of  $CAT$  subsets of?
  - (ii) Which element serves to index the members of  $CAT$ ?
  - (iii) Please represent  $CAT$  as an indexed family of sets.
  - (iv) Which elements in  $L$  work together to generate  $CAT$ ?

(2) **Basic Exercise in Language Design**

Let  $\langle A, F_\gamma, X_\delta, S_E, S \rangle_{\gamma \in \{\text{Merge, And, Not}\}, \delta \in \{\text{NP, IV, TV, S}\}}$  be the language ‘Mini-English’, as defined in (29) on the handout “An Algebraic Perspective on the Syntax of First Order Logic”.

- a. Please alter this structure minimally, so that the category  $C_S$  includes strings like the following:  
*If Mitt smokes, then Barack smokes.*
- b. Please provide an analysis tree showing how the following string is derived.  
*If Barack loves Michelle, then it is not the case that Mitt smokes.*
- c. Are the following meaningful expressions of the language you defined? Why or why not?
  - (i) *Then Barack smokes.*
  - (ii) *If Barack loves Michelle.*
  - (ii) *And Barack smokes.*

(3) **More Advanced Exercise on Language Design**

Let First Order Logic (FOL) be the language defined in (16) on the handout “An Algebraic Perspective on Propositional Logic.”

- a. Represent FOL as a language  $\langle A, F_\gamma, X_\delta, S, \delta_0 \rangle_{\gamma \in \Gamma, \delta \in \Delta}$ , following the definition in (21) on the handout “An Algebraic Perspective on the Syntax of First Order Logic”. To do this, you will need to do the following:

- (i) Identify a set of syntactic operations  $\{ F_\gamma \}_{\gamma \in \Gamma}$  that will generate the WFFs of FOL (as defined in (16)).
- (ii) Use these operations to define an algebra  $\langle A, F_\gamma \rangle_{\gamma \in \Gamma}$  such that the WFFs of FOL are a subset of A.
- (iii) **Special Hint:** Let the syntactic category labels  $\Delta$  be  $T \cup \{\text{var}\}$ , where T is the set of types, and ‘var’ is the category label for variables.
- (iv) Organize the basic expressions of FOL into sets  $\{ X_\delta \}_{\delta \in \Delta}$   
**Special Hint:** Make sure that  $X_e$  contains both the individual constants and the variables.
- (v) Write out a set of syntactic rules S for FOL.  
**Special Hint:** You *might* find it helpful to consult (17) on “An Algebraic Perspective on Propositional Logic”

- b. Given your representation of FOL as a language, please provide an analysis tree showing how the following formula is generated by your system (where P is a unary predicate letter and Q is a binary predicate letter):

$$\forall x(Px \ \& \ \sim \exists y((Qa)y))$$

- c. Let R be a ternary predicate letter, a and c be individual constants, and x be a variable.

- (i) Please state whether the following are or are not meaningful expressions of the language you defined.
  1.  $((Ra)x)$
  2.  $((Ra)x)c$
  3.  $\exists x((Ra)x)$
- (ii) If the string is a meaningful expression, please provide a calculation showing what category it is a member of (follow the format in (17) on the handout “An Algebraic Perspective on the Syntax of First Order Logic”).
- (iii) If the string is not a meaningful expression, please provide a brief explanation of why it isn’t.

(4) **Another Exercise on Language Design**

Let Propositional Logic (PL) be the language defined in (4) on the handout “An Algebraic Perspective on Propositional Logic.” Please represent PL as a language, following the definition in (21) on the handout “An Algebraic Perspective on the Syntax of First Order Logic”.

**HINT:** Adapt what you did in exercise (3).

(5) **An Exercise on Models and (Montagovian) Interpretations**

Let FOL-NoQ be the language  $\langle A, F_\gamma, X_\tau, S, t \rangle_{\gamma \in \{\text{Concat, Not, And}\}, \tau \in T}$  defined in (5) on the handout “Montague’s General Theory of Semantics” (MGTS).

- a. Let E be a set of entities, and let  $\mathbf{B} = \langle B, G_\gamma, f \rangle_{\gamma \in \{\text{Concat, Not, And}\}}$  be a Fregean interpretation for FOL-NoQ based on E (as defined in (15) on MGTS). Let  $\mathcal{M}$  be a model  $\langle E, I \rangle$  (as defined in (21) on MGTS), where  $I = f$ .

**Please show** via induction on structural complexity that every  $\varphi \in C_t$  is such that  $[[\varphi]]^M = g(\varphi)$ , where g is the meaning assignment based on  $\mathbf{B}$ .

**Some Hints:**

1. First, show that if  $\varphi$  is an atomic formula of FOL-NoQ, then  $[[\varphi]]^M = g(\varphi)$ .  
To show this, first use our definition of a model  $\mathcal{M}$  to show:  

$$[[ (\dots(\Phi\alpha_1)\dots\alpha_n) ]]^M = g(\Phi)(g(\alpha_1))\dots(g(\alpha_n))$$
Then, use the homomorphism property of g to show:  

$$g(\Phi)(g(\alpha_1))\dots(g(\alpha_n)) = g( (\dots(\Phi\alpha_1)\dots\alpha_n) )$$
2. Next, assume that  $\varphi$  is a conjunction ( $\psi \& \chi$ ), and that  $\psi$  and  $\chi$  are both such that  $[[\psi]]^M = g(\psi)$  and  $[[\chi]]^M = g(\chi)$ . Show that  $[[\varphi]]^M = g(\varphi)$ .  
To show this, use the definition of a model  $\mathcal{M}$  and the induction assumption to show:  

$$[[ (\psi \& \chi) ]]^M = G_{\text{And}}( g(\psi), g(\chi) )$$
Then use the homomorphism property of g to show:  

$$G_{\text{And}}( g(\psi), g(\chi) ) = g( (\psi \& \chi) )$$
3. Next, assume that  $\varphi$  is a negation  $\sim\psi$ , and that  $\psi$  is such that  $[[\psi]]^M = g(\psi)$ . Show that  $[[\varphi]]^M = g(\varphi)$ .  
To show this, follow the same general strategy laid out in 2.

- b. Let  $\mathcal{M}$  be a model  $\langle D, I \rangle$ . Let  $\mathbf{B} = \langle B, G_\gamma, f \rangle_{\gamma \in \{\text{Concat, Not, And}\}}$  be a Fregean interpretation for FOL-NoQ based on D, where  $f = I$ .

**Please show** via induction on structural complexity that every  $\varphi \in C_t$  is such that  $[[\varphi]]^M = g(\varphi)$ , where g is the meaning assignment based on  $\mathbf{B}$ .

**Unit 4:**  
**Montague's Theory of Translation**

## Montague's Theory of Translation: Laying the Groundwork<sup>1</sup>

### 1. Some Context and Questions to Have in the Background

#### (1) Fundamental Fact: Translation ≠ Semantics

- Translating expressions from a language  $L$  into a language  $L'$  doesn't *necessarily* tell you anything about the semantics of the expressions in language  $L$ .
  - For example, simply knowing that (1a) can be translated as (1b) doesn't tell you anything about what (1a) *means*.
- a. Tlingit Sentence:  $\underline{Ax}$  éet yaan uwaháa
- b. Haida Translation: Dii.uu q'wiidang gwaa.
- However, translating an expression from a language  $L$  into a language  $L'$  **whose semantics are known** *can* inform you of the semantics of the expressions in  $L$ .
  - For example, given that you speak English, knowing that (1a) can be translated as (1c) *would* inform you of the meaning of (1a).
- c. English Translation: I'm hungry. (~ Hunger moves to me imperceptibly.)
- Of course, even such translations don't necessarily mean one has a *compositional semantics* for language  $L$ .

#### (2) Question (Not Necessarily Montague's)

Given our background theory of language and meaning, under what conditions (if any) can we guarantee that translating from one language  $L$  into another language  $L'$  gives us a compositional semantics for  $L$ .

#### (3) Some Historical Context: Semantics in Generative Grammar Before Montague

Prior to Montague (and for some time afterwards), generative linguists conceived of natural language semantics as having the following goal:

- Develop a theory of the system that maps syntactic structures to some kind of 'mentalese' (conceptual structures) encoding the information in the sentence.

Illustration: Jackendoff's 'Conceptual Semantics'

"A dog is a reptile" → [State Is-included in ( [Thing Type: Dog] ) ( [Thing Type: Reptile] ) ]  
(Jackendoff 1983: 96)

<sup>1</sup> These notes are based upon material in the following readings: Halvorsen & Ladusaw (1979), Dowty *et al.* (1981) Chapter 8, and Thomason (1974) Chapter 7 (Montague's "Universal Grammar").

(4) **Early and Perennial Criticism of Such Approaches**

Given the fundamental fact in (1), simply translating a sentence of English to a sentence of ‘mentalese’ isn’t (necessarily) providing a semantics for the English sentence.

- The problem of providing a semantics for English now becomes the problem of providing a semantics for the ‘mentalese’ notation (one that had never been taken up)

(5) **Question (Not Necessarily Montague’s)**

Under what conditions can this problem in (4) be circumvented? Again, under what conditions (if any) can we guarantee that translating from one language  $L$  into another language (notation)  $L'$  gives us a compositional semantics for  $L$ .

## 2. Key Ingredient: First Order Logic as a *Family* of Languages

Up to now, we’ve been using the term “First Order Logic” to refer to a *single* language...

- However, at this point, it will be important to view First Order Logic not as a single language, but rather as a family of infinitely many different languages...

*First Order Logic A:*       $\exists x( (Px) \& ((Qa)b) )$

*First Order Logic B:*       $\exists x( (\text{dog' } x) \& ((\text{loves' bill' }) \text{ mary'}) )$

*First Order Logic C:*       $\exists x( (\mathcal{D} \star) \& ((\mathcal{P} \star) \mathbf{*}) )$

(6) **First Order Language (Logic)**

A first order language (first order logic) is a language whose vocabulary of symbols satisfies the conditions in (6a) and whose WFFs satisfy the conditions in (6b).

a. The Vocabulary of a First Order Language (Logic):

- The Logical Constants:*       $\sim, \&, \forall, (\vee, \rightarrow, \exists$  are ‘abbreviations’)
- Syntactic Symbols:*       $(, )$
- The Non-Logical Constants:*
  1. A **countable** set of predicate letters (with associated arities)
  2. A **countable** set of individual constants
  3. A **countably infinite** set of variables  $\{x, y, z, \dots, x_1, x_2, x_3, \dots\}$

b. The Well-Formed Formulas of a First Order Language (Logic):

- If  $\varphi$  is an n-ary predicate letter and each of  $\alpha_1, \dots, \alpha_n$  is either an individual constant or a variable, then  $\text{Concat}(\dots(\text{Concat}(\text{Concat}(\varphi, \alpha_1), \alpha_2), \dots, \alpha_n)) \in \text{WFF}$
- If  $\varphi, \psi \in \text{WFF}$ , then  $\sim\varphi \in \text{WFF}$  and  $(\varphi \& \psi) \in \text{WFF}$
- If  $\varphi \in \text{WFF}$  and  $v$  is a variable, then  $\forall v\varphi \in \text{WFF}$

(7) **Remarks**

- a. Note that a first order language only needs a *countable* set of predicate letters and individual constants. Thus, in a first order language, those sets can be *finite*.
- b. Note that we are requiring a first order language to use the vocabulary in (6ai, ii), the (infinite) variables in (iii), and the exact syntax rules in (6b).
  - Thus, a logic in ‘Polish notation’ wouldn’t be a FOL according to (6); nor would one making use of the alternate symbols ‘¬’ and ‘^’
  - The reason for this is simply because we need to keep *some* things constant between FOLs; it works for us right now to keep these constant.
- c. The definition in (6) defines an infinite set of different languages/logics.
  - We can think of the general term ‘First Order Logic’ as referring to this infinite set of different languages.

(8) **Illustration: The First Order Language ‘Politics’**

The language ‘Politics’ is the first order language whose vocabulary is as in (8a) and whose WFFs are defined in (8b).

- a. The Vocabulary of ‘Politics’
  - (i) *The Logical Constants:* ~, &,  $\forall$ ,
  - (ii) *Syntactic Symbols:* (, )
  - (iii) *The Non-Logical Constants:*
    - 1. Predicate Letters:
      - Unary Predicate Letters: { **smokes'** }
      - Binary Predicate Letters: { **loves'** }
    - 2. Individual Constants: { **michelle'**, **barack'**, **mitt'** }
    - 3. Variables: { x, y, z, ...,  $x_1, x_2, x_3, \dots$  }
- b. The WFFs of ‘Politics’ (exactly as in (6b))

(9) **Some Illustrative Formulae of Politics**

- a.  $\sim(\text{smokes}' \text{ barack}')$
- b.  $(\text{(loves}' \text{ barack}') \text{ michelle}')$
- c.  $\sim((\text{smokes}' \text{ barack}') \& (\text{smokes}' \text{ mitt}'))$
- d.  $\forall x \sim((\text{smokes}' x) \& \sim(\text{loves}' \text{ mitt}') x)$

Notice how the vocabulary in our logical language is boldfaced, and followed by primes?...  
*Get used to that!...*

### 3. Introducing the Central Characters

#### (10) A Roadmap of Where We're Headed

We're going to build up Montague's theory of translation by showing how a fragment of English can be 'translated' (formally) into a fragment of Politics, and then showing how that (formal) translation also gives us a semantics for the fragment of English.

- a. *Step One:* Build the relevant fragment of Politics
- b. *Step Two:* Try to build the relevant fragment of English
- c. *Step Three:* NOTICE A FUNDAMENTAL PROBLEM IN STEP TWO
- d. *Step Four:* Fix that problem, leading to a further refinement of our definition of what a language is...

*As we've done before, we're going to make our lives easier by putting aside quantification for the moment...*

#### (11) A Useful Fragment of Politics: Politics-NoQ

The language 'Politics-NoQ' is the language whose vocabulary is as in (11a) and whose WFFs are defined in (11b).

- a. The Vocabulary of 'Politics-NoQ'
  - (i) *The Logical Constants:*  $\sim$ , &
  - (ii) *Syntactic Symbols:* (, )
  - (iii) *The Non-Logical Constants:*
    1. Predicate Letters:
      - Unary Predicate Letters: { **smokes'** }
      - Binary Predicate Letters: { **loves'** }
    2. Individual Constants: { **michelle'**, **barack'**, **mitt'** }
- b. The WFFs of 'Politics-NoQ'
  - (i) If  $\varphi$  is an n-ary predicate letter and each of  $\alpha_1, \dots, \alpha_n$  is an individual constant, then  $\text{Concat}(\dots(\text{Concat}(\text{Concat}(\varphi, \alpha_1), \alpha_2), \dots, \alpha_n)) \in \text{WFF}$
  - (ii) If  $\varphi, \psi \in \text{WFF}$ , then  $\sim\varphi \in \text{WFF}$  and  $(\varphi \ \& \ \psi) \in \text{WFF}$

#### (12) Some Illustrative Formulae of Politics-NoQ

- a.  $\sim(\text{smokes'} \text{ barack'})$
- b.  $(\text{(loves'} \text{ barack')} \text{ michelle'})$
- c.  $\sim((\text{smokes barack'}) \ \& \ (\text{smokes'} \text{ mitt'}))$

(13) **Remarks**

- Politics-NoQ is *not* a ‘first order language’, as defined in (6).
- In terms of its structure, Politics-NoQ is quite similar to our language FOL-NoQ from the last two handouts.
- Consequently, we can easily see how to characterize Politics-NoQ in terms of our general (Montagovian) definition of a language.

(14) **The Language Politics-NoQ (Montagovian Presentation)**

The language Politics-NoQ is the structure  $\langle A, F_\gamma, X_\tau, S, t \rangle_{\gamma \in \{\text{Concat, Not, And}\}, \tau \in T}$  where:

- a.  $\langle A, F_\gamma \rangle_{\gamma \in \{\text{Concat, Not, And}\}}$  is the algebra such that
  - (i)  $F_{\text{Concat}}, F_{\text{Not}}, F_{\text{And}}$  are as defined previously.
  - (ii)  $A$  is the smallest set such that:
    1.  $\{ \text{'smokes'}, \text{'loves'}, \text{'michelle'}, \text{'barack'}, \text{'mitt'} \} \subseteq A$
    2. It is closed under  $F_{\text{Concat}}, F_{\text{Not}}, F_{\text{And}}$
- b. The basic categories  $X_\tau$  are such that:
  - (i)  $X_e = \{ \text{'michelle'}, \text{'barack'}, \text{'mitt'} \}$
  - (ii)  $X_{\langle e, t \rangle} = \{ \text{'smokes'} \}$
  - (iii)  $X_{\langle e, \langle e, t \rangle \rangle} = \{ \text{'loves'} \}$
  - (iv) For all other types  $\tau \in T$ ,  $X_\tau = \emptyset$ .
- c. The set  $S$  is the following (infinite) set of syntactic rules:
  - (i)  $\langle F_{\text{Not}}, \langle t \rangle, t \rangle$
  - (ii)  $\langle F_{\text{And}}, \langle t, t \rangle, t \rangle$
  - (iii)  $\langle F_{\text{Concat}}, \langle \langle \sigma, \tau \rangle, \sigma \rangle, \tau \rangle$ , for all  $\sigma, \tau \in T$

(15) **Some Illustrative Members of Category  $C_t$  of Politics-NoQ**

- a.  $\sim(\text{'smokes'} \text{'barack'})$
- b.  $((\text{'loves'} \text{'barack'}) \text{'michelle'})$
- c.  $\sim((\text{'smokes'} \text{'barack'}) \& (\text{'smokes'} \text{'mitt'}))$

*Given this structural similarity between Politics-NoQ and FOL-NoQ, it's also rather easy to set up a (Fregan) interpretation for Politics-NoQ!*

(16) A (Fregean) Interpretation of Politics-NoQ

Let the set  $S = \{ \text{Michelle}, \text{Barack}, \text{Mitt} \}$ . Let  $\mathbf{B} = \langle B, G_\gamma, f \rangle_{\gamma \in \{\text{Concat}, \text{Not}, \text{And}\}}$  be the Fregean interpretation based on  $S$ , such that  $f$  consists of the following mappings:

- a.  $f(\text{michelle}') = \text{Michelle}$
- b.  $f(\text{barack}') = \text{Barack}$
- c.  $f(\text{mitt}') = \text{Mitt}$
- d.  $f(\text{smokes}') = h = \{ \langle \text{Michelle}, 0 \rangle, \langle \text{Barack}, 1 \rangle, \langle \text{Mitt}, 0 \rangle \}$
- e.  $f(\text{loves}') = j = \begin{array}{ccc} \text{Michelle} & \rightarrow & \left\{ \begin{array}{l} \text{Michelle} \rightarrow 1 \\ \text{Barack} \rightarrow 1 \\ \text{Mitt} \rightarrow 0 \end{array} \right\} \\ \text{Barack} & \rightarrow & \left\{ \begin{array}{l} \text{Michelle} \rightarrow 1 \\ \text{Barack} \rightarrow 1 \\ \text{Mitt} \rightarrow 0 \end{array} \right\} \\ \text{Mitt} & \rightarrow & \left\{ \begin{array}{l} \text{Michelle} \rightarrow 0 \\ \text{Barack} \rightarrow 0 \\ \text{Mitt} \rightarrow 1 \end{array} \right\} \end{array}$

Note: We've basically interpreted ‘smokes’ as the property ‘smokes’, and we've interpreted ‘loves’ as the (curried) relation ‘ $x$  is loved by  $y$ ’.<sup>2</sup>

(17) Using the Fregean Interpretation in (16) to Interpret Sentences of Politics-NoQ

- (i)  $g(\sim(\text{smokes}' \text{ barack}')) = \text{ (by definition of Politics-NoQ)}$
- (ii)  $g(F_{\text{Not}}(F_{\text{Concat}}(\text{smokes}', \text{barack}')))) = \text{ (by homomorphism property of } g)$
- (iii)  $G_{\text{Not}}(g(F_{\text{Concat}}(\text{smokes}', \text{barack}')))) = \text{ (by homomorphism property of } g)$
- (iv)  $G_{\text{Not}}(G_{\text{Concat}}(g(\text{smokes}'), g(\text{barack}')))) = \text{ (by definition of } g)$
- (v)  $G_{\text{Not}}(G_{\text{Concat}}(f(\text{smokes}'), f(\text{barack}')))) = \text{ (by definition of } f \text{ in (16))}$
- (vi)  $G_{\text{Not}}(G_{\text{Concat}}(h, \text{Barack})) = \text{ (by definition of } G_{\text{Concat}})$
- (vii)  $G_{\text{Not}}(h(\text{Barack})) = \text{ (by definition of } h \text{ in (16d))}$
- (viii)  $G_{\text{Not}}(1) = \text{ (by definition of } G_{\text{Not}})$
- (ix)  $0$

---

<sup>2</sup> That is, in the notation of Heim & Kratzer (1998), we're interpreting ‘loves’ as  $[\lambda y: [\lambda x: x \text{ loves } y]]$ . The semanticists in the house can probably guess why we're doing this ; )

Finally, to get our third player on the field, let's recall that fragment of English that we defined a while back...

(18) **The Definition of ‘Mini-English’**

‘Mini-English’ is the structure  $\langle E, K_\gamma, X_\delta, S_E, S \rangle_{\gamma \in \{\text{Concat, Not, And}\}, \delta \in \Delta}$  such that:

a. The Syntactic Categories:  $\Delta = \{\text{NP, IV, TV, S}\}$

b. The Syntactic Operations:

- (i)  $K_{\text{Concat}} = \text{Merge}$  (from previous notes)
- (ii)  $K_{\text{Not}} = \text{Not}_E$  (from previous notes)
- (iii)  $K_{\text{And}} = \text{And}_E$  (from previous notes)

c. The Basic Expressions:

- (i)  $X_{\text{NP}} = \{\text{Barack, Michelle, Mitt}\}$
- (ii)  $X_{\text{IV}} = \{\text{smokes}\}$
- (iii)  $X_{\text{TV}} = \{\text{loves}\}$
- (iv)  $X_S = \emptyset$

c. The Syntactic Algebra:

$E$  is the smallest set such that:

- (i) For all  $\delta \in \Delta$ ,  $X_\delta \subseteq E$ .
- (ii)  $E$  is closed under the operations  $K_{\text{Concat}}$ ,  $K_{\text{Not}}$  and  $K_{\text{And}}$

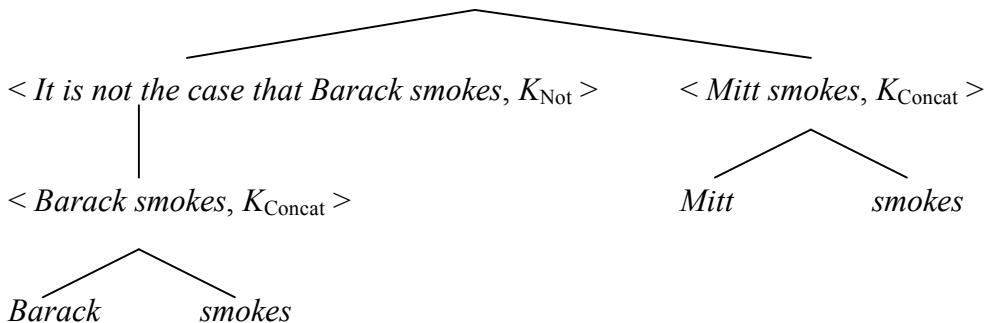
d. The Syntactic Rules: The set  $S_E$  consists of the following tuples:

- (i)  $\langle K_{\text{Concat}}, \langle \text{TV, NP} \rangle, \text{IV} \rangle$
- (ii)  $\langle K_{\text{Concat}}, \langle \text{NP, IV} \rangle, \text{S} \rangle$
- (iii)  $\langle K_{\text{And}}, \langle \text{S, S} \rangle, \text{S} \rangle$
- (iv)  $\langle K_{\text{Not}}, \langle \text{S} \rangle, \text{S} \rangle$

(19) **Illustrative Sentence (Expression of Category Cs) of Mini-English**

*It is not the case that Barack smokes and Mitt smokes.*

$\langle \text{It is not the case that Barack smokes and Mitt smokes}, K_{\text{And}} \rangle$

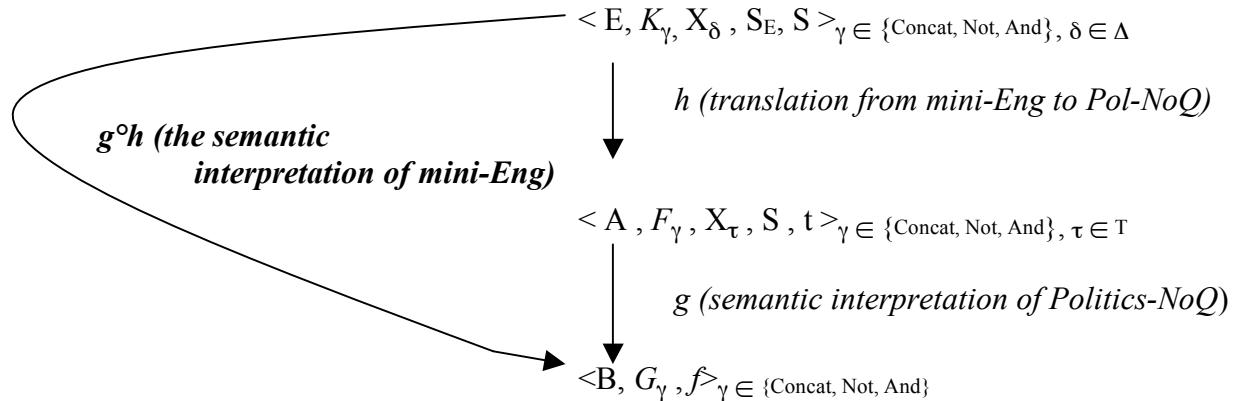


Unfortunately, there's a fundamental problem with the language as defined in (18). To see this, recall our ultimate goal, informally sketched out below.

(20) **Our Goal for a Theory of Translation**

We want to develop a way of homomorphically mapping expressions of mini-English to expressions of Politics-NoQ (so that we can ultimately get a semantics for English)

Indirect Interpretation in a Picture (Oversimplified):

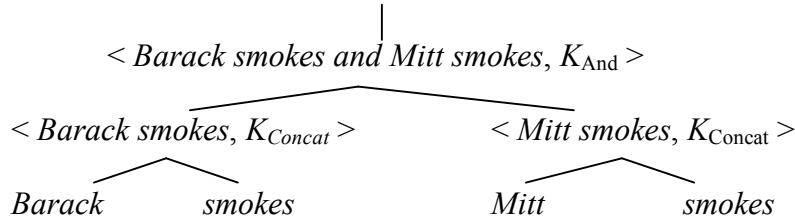


(21) **Critical Problem**

We currently conceive of the set E (expressions of mini-English) as consisting of *strings of English words*.

- However, some such strings in our mini-English language can be created **in more than one way** from the syntactic operations (and rules) of our language.

Example:     $\langle \text{It is not the case that } \text{Barack smokes and Mitt smokes}, K_{\text{Not}} \rangle$



- Intuitively, we want these two different ways of constructing the mini-English sentence to lead to two different Politics-NoQ translations:
 
$$\begin{aligned}
 & (\sim(\text{smokes' barack'}) \& (\text{smokes' mitt'})) \\
 & \sim(\text{smokes' barack'}) \& (\text{smokes' mitt'})
 \end{aligned}$$
- But if the translation h is a mapping from strings of English to expressions of Politics-NoQ, each such string will be mapped to only one translation!**

### (22) Some More General Remarks

- Ultimately, we want there to be two different Politics-NoQ translations of sentence (19) because we want this string to be paired with two different semantic values.
  - *There is a reading where (19) is true*  
 $\approx \sim(\text{smokes' barack'}) \& (\text{smokes' mitt'})$
  - *There is a reading where (19) is false*  
 $\approx (\sim(\text{smokes' barack'}) \& (\text{smokes' mitt'}))$
- Also, we have the background belief that (19) has these two readings *because of* the different ways that the sentence can be constructed in English
  - (i.e., it's not because of any ambiguity in what the words mean...)
- But, if we are semantically interpreting *strings* of English words – and ‘interpretation’ is conceived us as a *homomorphism* (function) from expressions to meanings – then each string will be mapped to exactly one meaning.
- Thus, unlike with Politics-NoQ, it is not feasible to build a semantics that interprets (directly or indirectly) English *strings*.
- In LING 610, this problem doesn’t even arise, because right from the start we’re interpreting phrase structure trees
  - After all, a given tree is only ever constructed in one way by Merge and Move... *hmm...*

### (23) Another Critical Problem

In the picture in (20), translation is a homomorphism from  $\langle E, K_\gamma \rangle_{\gamma \in \{\text{Concat}, \text{Not}, \text{And}\}}$  for mini-English to the algebra  $\langle A, F_\gamma \rangle_{\gamma \in \{\text{Concat}, \text{Not}, \text{And}\}}$  for Politics-NoQ

- Under such a homomorphism, we’d naturally want  $K_{\text{Concat}}$  and  $F_{\text{Concat}}$  to correspond. This will get the right interpretation for VPs (IVs) after all:

$$h(\text{ loves michelle }) = h(K_{\text{Concat}}(\text{ loves }, \text{ Michelle })) = \\ F_{\text{Concat}}(h(\text{loves}), h(\text{Michelle})) = F_{\text{Concat}}(\text{loves}', \text{michelle}') = (\text{loves}' \text{ michelle}')$$

- **However, this will get the wrong result for sentences!** Sentences will end up mapped to syntactic garbage in A.

$$h(\text{ Barack smokes }) = h(K_{\text{Concat}}(\text{ Barack }, \text{ smokes })) = \\ F_{\text{Concat}}(h(\text{Barack}), h(\text{smokes})) = F_{\text{Concat}}(\text{barack}', \text{smokes}') = (\text{barack}' \text{ smokes}')$$

**Syntactic Garbage!!!**

(24) More General Remarks

- Obviously, what we want is for  $h(\text{Barack smokes}) = (\text{smokes' barack}')$
- But, there is no operation in the algebra for Politics-NoQ which will take as argument the translation of *Barack* (**barack'**) and the translation of *smokes* (**smokes'**) and return the formula (**smokes' barack'**)
- *So maybe our homomorphic translation function h shouldn't actually be a homomorphism to the algebra*  $\langle A, F_\gamma \rangle_{\gamma \in \{\text{Concat, Not, And}\}}$ 
  - Maybe it should be a homomorphism to some *other* algebra that we can construct from  $\langle A, F_\gamma \rangle_{\gamma \in \{\text{Concat, Not, And}\}}$

In these notes, we'll deal only with the problem in (21)-(22)...

In the next set of notes, we'll tackle the problem in (23)-(24)...

---

4. Montague's Notion of a 'Disambiguated Language'

(25) What We Want

- We want it to be that the interpreted expressions of our language can only ever be created from the syntactic operations (rules) in exactly one way.
- This way, we won't ever have to worry about interpreting 'syntactically ambiguous' expressions (*because they just won't exist in our language*).
- Preview of Where This is Going:  
We'll relate such 'syntactically unambiguous' expressions to sentence strings of English via a special operation (akin to 'linearization' or 'Spell Out').

(26) Question

Below we have our earlier (Montagovian) definition of a language. ***What do we have to add to this to ensure that no expressions are syntactically ambiguous?***

A language  $L$  is a structure  $\langle A, F_\gamma, X_\delta, S, \delta_0 \rangle_{\gamma \in \Gamma, \delta \in \Delta}$  such that:

- a.  $\langle A, F_\gamma \rangle_{\gamma \in \Gamma}$  is an algebra.
- b.  $A$  is the smallest set such that:
  - (i) For all  $\delta \in \Delta$ ,  $X_\delta \subseteq A$ ; (ii)  $A$  is closed under the operations  $F_\gamma$  for all  $\gamma \in \Gamma$
- c.  $S$  is a set of sequences of the form  $\langle F_\gamma, \langle \delta_1, \dots, \delta_n \rangle, \delta \rangle$ , where  $\gamma \in \Gamma$ ,  $F_\gamma$  is an  $n$ -ary operation, and  $\delta_1, \dots, \delta_n, \delta \in \Delta$
- b.  $\delta_0 \in \Delta$

(27) **Montague's Answer**

At a minimum, we need to ensure that:

- a. Nothing in the basic expressions  $X_\delta$  (lexical items) can also be constructed by the syntactic operations.

That is:  $X_\delta$  and the range of  $F_\gamma$  are disjoint for all  $\delta \in \Delta$  and  $\gamma \in \Gamma$

- b. No element of  $A$  will be the output of two different operations  $F_\gamma$  and  $F_{\gamma'}$ ,  
c. No single operation  $F_\gamma$  will take two different inputs  $a, a' \in A$  and give the same output.

That is: For all sequences  $a_1, \dots, a_n \in A^n$  and  $a'_1, \dots, a'_m \in A^m$ , if  $F_\gamma(a_1, \dots, a_n) = F_{\gamma'}(a'_1, \dots, a'_m)$ , then  $F_\gamma = F_{\gamma'}$  and  $\langle a_1, \dots, a_n \rangle = \langle a'_1, \dots, a'_m \rangle$ .

Note that if the conditions in (27) hold, then every expression in  $A$  will either be (i) a basic expression (lexical item), or (ii) constructible in exactly one way from the syntactic operations.

(28) **Montagovian Definition of a 'Diambiguated Language'**

The following definition now replaces our earlier concept of a language, as well as its concomitant definition.

A *disambiguated language* is a structure  $\langle A, F_\gamma, X_\delta, S, \delta_0 \rangle_{\gamma \in \Gamma, \delta \in \Delta}$  such that:

- a.  $\langle A, F_\gamma \rangle_{\gamma \in \Gamma}$  is an algebra.
- b.  $A$  is the smallest set such that:  
(i) For all  $\delta \in \Delta$ ,  $X_\delta \subseteq A$ ; (ii)  $A$  is closed under the operations  $F_\gamma$  for all  $\gamma \in \Gamma$
- c.  $X_\delta$  and the range of  $F_\gamma$  are disjoint for all  $\delta \in \Delta$  and  $\gamma \in \Gamma$
- d. For all sequences  $a_1, \dots, a_n \in A^n$  and  $a'_1, \dots, a'_m \in A^m$ , if  $F_\gamma(a_1, \dots, a_n) = F_{\gamma'}(a'_1, \dots, a'_m)$ , then  $F_\gamma = F_{\gamma'}$  and  $a_1, \dots, a_n = a'_1, \dots, a'_m$ .
- e.  $S$  is a set of sequences of the form  $\langle F_\gamma, \langle \delta_1, \dots, \delta_n \rangle, \delta \rangle$ , where  $\gamma \in \Gamma$ ,  $F_\gamma$  is an  $n$ -ary operation, and  $\delta_1, \dots, \delta_n, \delta \in \Delta$
- f.  $\delta_0 \in \Delta$

(29) **Remarks**

- a. Our language Politics-NoQ is such a disambiguated language.
- b. Our language ‘mini-English’ is *not* a disambiguated language.

c. Potential Problem:

If we assume that the expressions of mini-English (and English) are strings, then we just aren’t going to be able to represent those systems as disambiguated languages.

d. Solution:

Along with the concept of a ‘disambiguated language’ in (28), we need a more general concept of a ‘language’.

(30) **Montagovian Definition of a Language (Final Version)**

A language is a pair  $\langle L, R \rangle$ , where  $L = \langle A, F_\gamma, X_\delta, S, \delta_0 \rangle_{\gamma \in \Gamma, \delta \in \Delta}$  is a disambiguated language, and  $R$  is a binary relation whose domain is  $A$ .

- This relation  $R$  is an ‘ambiguating relation’.
  - It maps expressions in the disambiguated language  $L$  to expressions *not necessarily from  $L$* .
- Importantly,  $R$  can be many-to-one (surjection), and so we can have more than one expression from the disambiguated language being mapped to the *same* expression in the range of  $R$  (hence, the term ‘ambiguating’)

(31) **Illustration: Politics-NoQ-SansParens**

a. Informal Definition

- (i) Vocabulary: *Same as Politics-NoQ*
- (ii) The WFFs of ‘Politics-NoQ-SansParens’
  - 1. If  $\varphi$  is an n-ary predicate letter and each of  $\alpha_1, \dots, \alpha_n$  is either an individual constant or a variable, then  $\varphi\alpha_1\dots,\alpha_n \in \text{WFF}$
  - 2. If  $\varphi, \psi \in \text{WFF}$ , then  $\sim\varphi \in \text{WFF}$  and  $\varphi \& \psi \in \text{WFF}$

*Illustrative Formulae:*

$\sim \text{smokes' barack'}$   
 $\text{loves' barack' michelle'}$   
 $\sim \text{smokes barack' \& smokes' mitt'}$

b. Formal Definition

The pair  $\langle \langle A, F_\gamma, X_\tau, S, t \rangle_\gamma \in \{\text{Concat, Not, And}\}, \tau \in T, R \rangle$ , where the structure  $\langle A, F_\gamma, X_\tau, S, t \rangle_\gamma \in \{\text{Concat, Not, And}\}, \tau \in T$  is Politics-NoQ, and  $R$  is the function that takes any element of  $A$  and deletes every parenthesis.

$$R(\sim((\text{smokes barack'}) \& (\text{smokes' mitt'}))) = \\ \sim \text{smokes barack' \& smokes' mitt'}$$

(32) **Remark**

Every disambiguated language  $\langle A, F_\gamma, X_\delta, S, \delta_0 \rangle_{\gamma \in \Gamma, \delta \in \Delta}$  can also be represented as a language  $\langle \langle A, F_\gamma, X_\delta, S, \delta_0 \rangle_{\gamma \in \Gamma, \delta \in \Delta}, R \rangle$

- Simply let R be the identity function!!

### (33) Some Concomitant Definitions

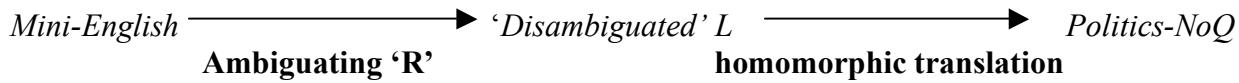
Let  $\mathbf{L}$  be a language  $\langle\langle A, F_\gamma, X_\delta, S, \delta_0 \rangle\rangle_{\gamma \in \Gamma, \delta \in \Delta}, R \rangle$ .

- a. The *proper expressions* of **L** is the range of R.
  - b. The *operation indices* of **L** is  $\Gamma$ .
  - c. The *category labels* of **L** is  $\Delta$
  - d. The *syntactic rules* of **L** is S.
  - e. The *basic expressions* of **L** of category  $\delta$  is  $\{ \varphi : \exists \psi \in X_\delta \text{ such that } \psi R \varphi \}$
  - f. The *category*  $\delta$  of **L** is  $\{ \varphi : \exists \psi \in C_\delta \text{ such that } \psi R \varphi \}$ , where  $C_\delta$  is in the family of categories generated by  $\langle A, F_\gamma, X_\delta, S, \delta_0 \rangle_{\gamma \in \Gamma, \delta \in \Delta}$
  - g. The *meaningful expressions* of **L** is the union of all the categories  $\delta$  of **L**.
  - h. The *declarative sentences* of **L** is the category  $\delta_0$  of **L**
  - i. If  $\varphi$  is a meaningful expression of **L**, then  $\varphi$  is *syntactically ambiguous* if there are distinct  $\psi, \psi' \in \cup_{\delta \in \Delta} C_\delta$  such that  $\psi R \varphi$  and  $\psi' R \varphi$
  - j. The language **L** is *syntactically ambiguous* if there is a meaningful expression  $\varphi$  of **L** which is syntactically ambiguous.

### (34) New Goal

Given all that we've laid out, it seems that we now want to do the following:

- a. Represent mini-English as a (syntactically ambiguous) language  $\langle L, R \rangle$ , where
    - (i)  $L$  is some syntactically unambiguous language, and
    - (ii)  $R$  can ‘transform’ expressions of  $L$  into expressions of mini-English.
  - b. Translate mini-English into Politics-NoQ *indirectly*, via translation from  $L$  into Politics-NoQ.



## 5. Representing Mini-English Via a Disambiguated Language

### (35) Key Question

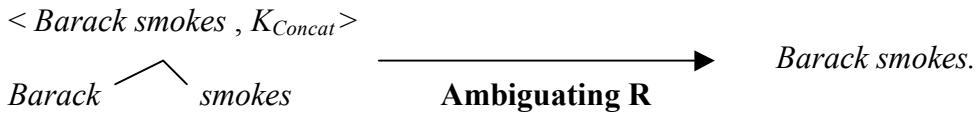
Given our new goal in (34), what should the expressions of our ‘disambiguated mini-English’ look like?

- Well, each complex expression must transparently reflect how it was constructed by the syntactic operations...
- That is, for each complex expression, there should be exactly one analysis tree...

### (36) Montague’s Core Insight

For a syntactically ambiguous natural language like English, we could assume that the syntactically disambiguated expressions *are the analysis trees themselves!!*

- That is, (mini-)English is a pair  $\langle L, R \rangle$ , where the expressions of  $L$  are analysis trees, and the relation  $R$  simply maps an analysis tree to the string in its root node!



### (37) Remarks

We’ll see in a moment how to actually implement the idea in (36). For the moment, let’s notice the similarities and differences between this and an ‘LF’-based semantics.

#### a. Key Similarity:

Our semantics does not directly interpret surface strings of English. Rather, it interprets abstract structures that represent how those strings can be derived.

#### b. Key Difference:

Unlike an ‘LF’-based semantics (like in 610), our system doesn’t first construct the analysis tree (LF structure) for a whole sentence and then ‘input’ that into semantic interpretation...

- That is, as will be clear in a few more classes, the syntax and semantics work in tandem with one another...

- Every time a ‘move’ is made in the syntax to make a structure, a corresponding ‘move’ is made in the semantics to determine a meaning for that structure...

*But how do we construct a ‘disambiguated language’ where the expressions are analysis trees?*

(38) **Step One: The Category Labels**

The syntactic categories of Disambiguated mini-English will be just the same as before:

$$\Delta = \{\text{NP}, \text{IV}, \text{TV}, \text{S}\}$$

(39) **The Basic Expressions**

The basic expressions of Disambiguated mini-English will be ‘trivial trees’. The following trees consisting of root-nodes without any daughters.

- a.  $X_{\text{NP}} = \{ <\text{Barack}, \emptyset>, <\text{Michelle}, \emptyset>, <\text{Mitt}, \emptyset> \}$
- b.  $X_{\text{IV}} = \{ <\text{smokes}, \emptyset> \}$
- c.  $X_{\text{TV}} = \{ <\text{loves}, \emptyset> \}$
- d.  $X_{\text{S}} = \emptyset$

(40) **The Syntactic Operations**

Our syntactic operations now take trees (including ‘trivial trees’) as input and output other trees, as defined below.

- In the definitions below,  $\alpha$  and  $\beta$  are trees whose root nodes are ordered pairs. In addition  $\alpha'$  and  $\beta'$  are the first members of the root nodes of  $\alpha$  and  $\beta$  (respectively).

a.  $K_{\text{Concat}}(\alpha, \beta) = <\alpha' \beta', \text{Concat}>$

```

graph TD
    K_Concat["K_Concat(\alpha, \beta)"] --- Concat["<α' β', Concat>"]
    Concat --- alpha["α"]
    Concat --- beta["β"]
  
```

b.  $K_{\text{Not}}(\alpha) = <\text{it is not the case that } \alpha', \text{Not}>$

```

graph TD
    K_Not["K_Not(α)"] --- Not["<it is not the case that α', Not>"]
    Not --- alpha["α"]
  
```

c.  $K_{\text{And}}(\alpha, \beta) = <\alpha' \text{ and } \beta', \text{And}>$

```

graph TD
    K_And["K_And(α, β)"] --- And["<α' and β', And>"]
    And --- alpha["α"]
    And --- beta["β"]
  
```

*Just for fun – since it will set us up for something important later, let’s also add the following syntactic operation.*

d.  $K_{\text{If}}(\alpha, \beta) = <\text{If } \alpha' \text{ then } \beta', \text{If}>$

```

graph TD
    K_If["K_If(α, β)"] --- If["<If α' then β', If>"]
    If --- alpha["α"]
    If --- beta["β"]
  
```

**Note:**  
The right-hand member of a node is now the **index of the operation**, rather than the operation itself...

(41) **The Syntactic Algebra**

E is the smallest set such that:

- a. For all  $\delta \in \Delta$ ,  $X_\delta \subseteq E$ .
- b. E is closed under the operations  $K_{\text{Concat}}$ ,  $K_{\text{Not}}$ ,  $K_{\text{And}}$ , and  $K_{\text{If}}$

(42) **The Syntactic Rules**

We can retain much the same set of syntactic rules  $S_E$  that we had before:

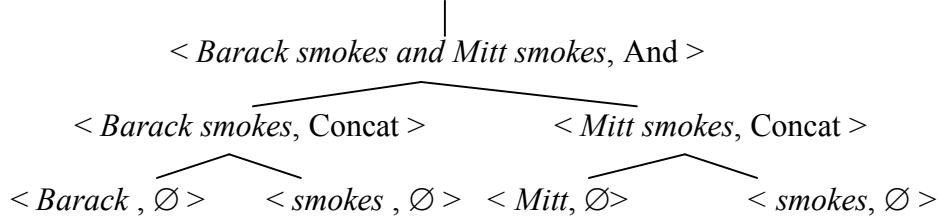
- a.  $\langle K_{\text{Concat}}, \langle TV, NP \rangle, IV \rangle$
- b.  $\langle K_{\text{Concat}}, \langle NP, IV \rangle, S \rangle$
- c.  $\langle K_{\text{And}}, \langle S, S \rangle, S \rangle$
- d.  $\langle K_{\text{If}}, \langle S, S \rangle, S \rangle$
- e.  $\langle K_{\text{Not}}, \langle S \rangle, S \rangle$

(43) **The Definition of Our Language: ‘Disambiguated Mini-English’**

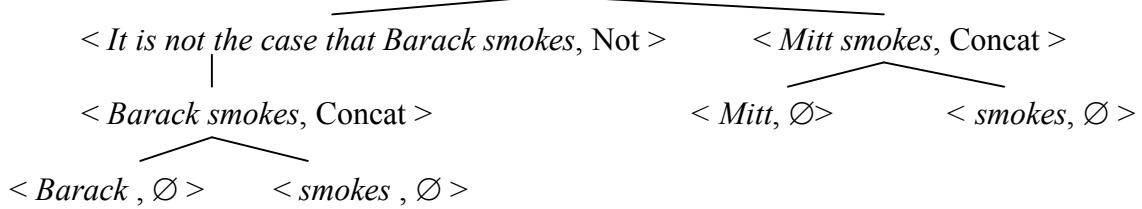
The structure  $\langle E, K_\gamma, X_\delta, S_E, S \rangle_\gamma \in \{\text{Concat, Not, And, If}\}, \delta \in \Delta$  where  $E, K_\gamma, X_\delta, S_E$ , and  $\Delta$  are as defined in (38)-(42).

Some Illustrative Members of Category  $C_S$

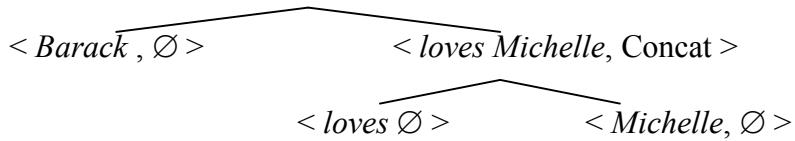
- a.  $\langle It \text{ is not the case that } Barack \text{ smokes and Mitt smokes}, \text{Not} \rangle$



- b.  $\langle It \text{ is not the case that } Barack \text{ smokes and Mitt smokes}, \text{And} \rangle$



- c.  $\langle Barack loves Michelle, \text{Concat} \rangle$



(44) **Remark** Disambiguated Mini-English is indeed a disambiguated language.

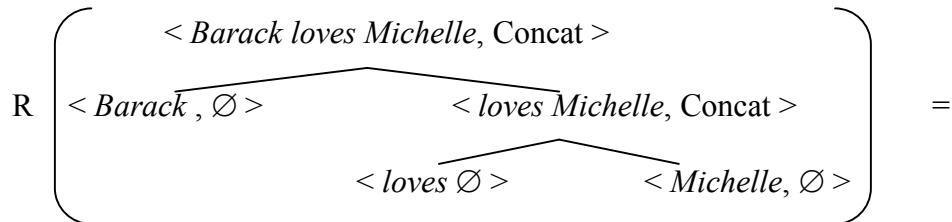
- No syntactic operation will ever create a basic expression.
- Because of the way the trees are indexed, no two ops will ever have the same output

Now, we can use Disambiguated Mini-English to characterize Mini-English as a language, in the sense of (30)

(45) **Montagovian Definition of Mini-English**

Mini-English is the structure  $\langle \langle E, K_\gamma, X_\delta, S_E, S \rangle_\gamma \in \{\text{Concat, Not, And, If}\}, \delta \in \Delta, R \rangle$ , where

- The structure  $\langle \langle E, K_\gamma, X_\delta, S_E, S \rangle_\gamma \in \{\text{Concat, Not, And, If}\}, \delta \in \Delta \rangle$  is Disambiguated Mini-English, as defined in (43).
- $R$  is a function which takes as input a tree  $T$  in  $E$ , and returns as output the first member of the root node of  $T$ .



Barack loves Michelle

(46) **Some Illustrative Members of the Category S for Mini-English**

- Barack smokes.*
- Barack loves Michelle.*
- It is not the case that Barack smokes and Mitt smokes.*

(47) **Remark** Mini-English is a syntactically ambiguous language (33j)

- After all, let  $T$  be the tree in (43a), and  $T'$  be the tree in (43b).
- $R(43a) = R(43b) = \text{It is not the case that Barack smokes and Mitt smokes.}$

**What Coming Up Next:**

- We now have the following two disambiguated languages:
  - Politics-NoQ
  - Disambiguated Mini-English
- We have an interpretation for Politics-NoQ
- **We're now going to try to find a way of homomorphically mapping expressions of Disambiguated Mini-English to ones of Politics-NoQ**

## Montague's Theory of Translation: The Notion of a 'Translation Base'<sup>1</sup>

### 1. Review of What We Have and Where We Want to Go

#### (1) Our Key Ingredients

##### a. Two Disambiguated Languages:

Both the languages below are such that every expression in the language is either (i) a basic expression, or (ii) formed in exactly one way from the syntactic operations, but (iii) *not both*.

##### (i) *Politics-NoQ*

The structure  $\langle A, F_\gamma, X_\tau, S, t \rangle_\gamma \in \{\text{Concat, Not, And}\}$ ,  $\tau \in T$  where the algebra  $\langle A, F_\gamma \rangle_\gamma \in \{\text{Concat, Not, And}\}$ , and the sets  $X_\tau$  and  $S$  are as before.

##### (ii) *Disambiguated Mini-English (DME)*

The structure The structure  $\langle E, K_\gamma, X_\delta, S_E, S \rangle_\gamma \in \{\text{Concat, Not, And, If}\}$ ,  $\delta \in \Delta$ , where algebra  $\langle E, K_\gamma \rangle_\gamma \in \{\text{Concat, Not, And, If}\}$ , and the sets  $X_\delta$ ,  $S_E$  as before.

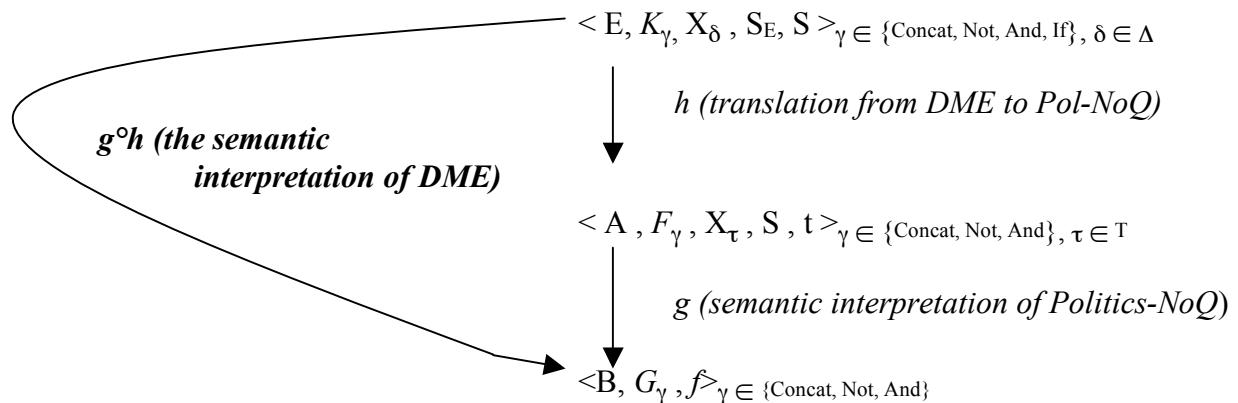
##### b. An Interpretation for Politics-NoQ

Let the set  $S = \{ \text{Michelle, Barack, Mitt} \}$ . Let  $\mathbf{B} = \langle B, G_\gamma, f \rangle_\gamma \in \{\text{Concat, Not, And}\}$  be the Fregean interpretation based on  $S$ , such that  $f$  is as defined before.

#### (2) What We Want to Make from Those Ingredients

We want to develop a way of homomorphically mapping expressions of DME to expressions of Politics-NoQ (so that we can ultimately get a semantics for English)

##### Indirect Interpretation in a Picture (Oversimplified):



<sup>1</sup> These notes are based upon material in the following readings: Halvorsen & Ladusaw (1979), Dowty *et al.* (1981) Chapter 8, and Thomason (1974) Chapter 7 (Montague's "Universal Grammar").

Although we've come closer to our goal by restricting our attention to 'disambiguated languages', there are still two key problems facing our project...

(3) **Critical Problem 1:**

- If translation  $h$  is to be a homomorphism from DME to a syntactic algebra for P-NoQ, then there must be a syntactic operation  $OP$  in the latter that 'corresponds' to  $K_{Concat}$ .
- Moreover, under this correspondence, it must be that:

$$\begin{aligned} h(K_{Concat}(\text{Barack}, \text{smokes})) &= \\ OP(h(\text{Barack}), h(\text{smokes})) &= \\ OP(\text{barack}', \text{smokes}') &= (\text{smokes}' \text{ barack}') \end{aligned}$$

$$\begin{aligned} h(K_{Concat}(\text{loves}, \text{Barack})) &= \\ OP(h(\text{loves}), h(\text{Barack})) &= \\ OP(\text{loves}', \text{Barack}') &= (\text{loves}' \text{ barack}') \end{aligned}$$

- **But this seems inconsistent!** How can  $OP(\text{barack}', \text{smokes}') = (\text{smokes}' \text{ barack}')$ , while  $OP(\text{loves}', \text{barack}') = (\text{loves}' \text{ barack})$ ???

(4) **Critical Problem 2 (New):**

- Our syntactic algebra for DME contains the operation  $K_{If}$ .
- Again, if  $h$  is to be a homomorphism from DME to a syntactic algebra for P-NoQ, there must be some syntactic operation in the latter that 'corresponds' to  $K_{If}$
- **But there isn't any!**

The Plan:

We'll go halfway to fixing the problem in (3); at which point, the problems in (3) and (4) will become the same. Then we'll solve that more general problem by introducing a new, central idea of Montague's: **the Translation Base**.

---

**2. Prolegomena: A Slight Change to Our Definition of Disambiguated Mini-English**

*We're going to introduce a slight change to our definition of DME...  
It will seem ad hoc for now, but we'll see independent motivation later on (with quantification)...*

(5) **Step One: The Category Labels**

The syntactic categories of Disambiguated Mini-English will be just the same as before:

$$\Delta = \{\text{NP}, \text{IV}, \text{TV}, \text{S}\}$$

## (6) The Basic Expressions

The basic expressions of DME will be just the same as before, too.

- a.  $X_{NP} = \{ <Barack, \emptyset>, <Michelle, \emptyset>, <Mitt, \emptyset> \}$
- b.  $X_{IV} = \{ <smokes, \emptyset> \}$
- c.  $X_{TV} = \{ <loves, \emptyset> \}$
- d.  $X_S = \emptyset$

## (7) The Syntactic Operations

Our set of syntactic operations for DME is going to be altered. We're going to split  $K_{Concat}$  into two different operations:  $K_{Merge-S}$  and  $K_{Merge-IV}$ .

- In the definitions below,  $\alpha$  and  $\beta$  are trees whose root nodes are ordered pairs. In addition  $\alpha'$  and  $\beta'$  are the first members of the root nodes of  $\alpha$  and  $\beta$  (respectively).

- a. 
$$K_{Merge-S}(\alpha, \beta) = \begin{array}{c} < \alpha' \beta', \text{Merge-S} > \\ \diagdown \quad \diagup \\ \alpha \qquad \beta \end{array}$$
- b. 
$$K_{Merge-IV}(\alpha, \beta) = \begin{array}{c} < \alpha' \beta', \text{Merge-IV} > \\ \diagdown \quad \diagup \\ \alpha \qquad \beta \end{array}$$
- c. 
$$K_{Not}(\alpha) = \begin{array}{c} < it \text{ is not the case that } \alpha', \text{Not} > \\ | \\ \alpha \end{array}$$
- d. 
$$K_{And}(\alpha, \beta) = \begin{array}{c} < \alpha' \text{ and } \beta', \text{And} > \\ \diagdown \quad \diagup \\ \alpha \qquad \beta \end{array}$$
- e. 
$$K_{If}(\alpha, \beta) = \begin{array}{c} < \text{If } \alpha' \text{ then } \beta', \text{If} > \\ \diagdown \quad \diagup \\ \alpha \qquad \beta \end{array}$$

*Right now, the only difference between  $K_{Merge-S}$  and  $K_{Merge-IV}$  is the index on the root of the output. Again, later on these operations will become more substantively different.*

## (8) The Syntactic Algebra

$E$  is the smallest set such that:

- a. For all  $\delta \in \Delta$ ,  $X_\delta \subseteq E$ .
- b.  $E$  is closed under the operations  $K_{Merge-S}$ ,  $K_{Merge-IV}$ ,  $K_{Not}$ ,  $K_{And}$ , and  $K_{If}$

With the changes to our syntactic operations in (7) come some concomitant changes to our syntactic rules...

### (9) The Syntactic Rules

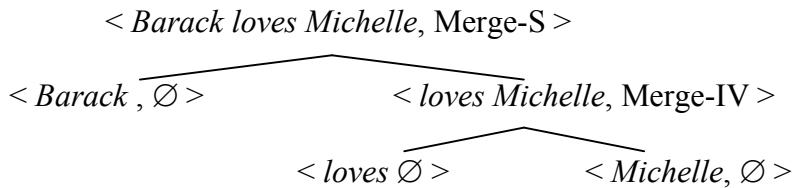
We can retain much the same set of syntactic rules  $S_E$  that we had before:

- a.  $\langle K_{\text{Merge-IV}}, \langle TV, NP \rangle, IV \rangle$
- b.  $\langle K_{\text{Merge-S}}, \langle NP, IV \rangle, S \rangle$
- c.  $\langle K_{\text{And}}, \langle S, S \rangle, S \rangle$
- d.  $\langle K_{\text{If}}, \langle S, S \rangle, S \rangle$
- e.  $\langle K_{\text{Not}}, \langle S \rangle, S \rangle$

### (10) New Definition for Disambiguated Mini-English

The structure  $\langle E, K_\gamma, X_\delta, S_E, S \rangle_\gamma \in \{\text{Merge-S}, \text{Merge-IV}, \text{Not}, \text{And}, \text{If}\}$ ,  $\delta \in \Delta$  where  $E, K_\gamma, X_\delta, S_E$ , and  $\Delta$  are as defined in (38)-(42).

Illustrative Structure:



### (11) Half of Problem (3) Solved

Now we don't need to find a single operation in Politics-NoQ that corresponds to  $K_{\text{Concat}}$

- In particular, we can now assume without problem that  $K_{\text{Merge-IV}}$  corresponds to  $F_{\text{Concat}}$
- |  |   |                                       |
|--|---|---------------------------------------|
| $h(K_{\text{Merge-IV}}( loves, Michelle ))$            | = |                                       |
| $F_{\text{Concat}}( h(loves), h(Michelle) )$           | = |                                       |
| $F_{\text{Concat}}( \text{loves}', \text{michelle}' )$ | = | $( \text{loves}' \text{ michelle}' )$ |

### (12) Remaining Problem

- Now, however, we have this additional operation  $K_{\text{Merge-S}}$ . **And, there doesn't seem to be an operation in Politics-NoQ that corresponds to it.**
- **And, we still need to find an operation in Politics-NoQ that corresponds to  $K_{\text{If}}$ !**

### (13) Key Idea Behind the Solution to (12)

Because of  $K_{\text{If}}$  and  $K_{\text{Merge-S}}$ , there won't be a homomorphism between the syntactic algebras  $\langle E, K_\gamma \rangle_\gamma \in \{\text{Merge-S}, \text{Merge-IV}, \text{Not}, \text{And}, \text{If}\}$  and  $\langle A, F_\gamma \rangle_\gamma \in \{\text{Concat}, \text{Not}, \text{And}\}$

- **But maybe we can make a new algebra from  $\langle A, F_\gamma \rangle_\gamma \in \{\text{Concat}, \text{Not}, \text{And}\}$  that will fit the bill!!!**

### 3. Polynomial Operations and Derived Syntactic Rules

#### (14) Road Map for This Section

In this section, we will see how to do the following:

- a. From an algebra  $\langle A, F_\gamma \rangle_{\gamma \in \Gamma}$ , create new complex operations from the operations  $\{ F_\gamma \}_{\gamma \in \Gamma}$ .
- b. From a (disambiguated) language  $\langle A, F_\gamma, X_\delta, S, \delta_0 \rangle_{\gamma \in \Gamma, \delta \in \Delta}$ , create new complex syntactic rules from the rules  $S$ .

This will provide us with the key tools for introducing Montague's notion of a 'translation base'.

#### Advisory:

In my own view, the technical concept of a 'translation base' is the least *a priori* intuitive ingredient of Montague's theory...

- Once you see what the thing does – and how it does what it does – it becomes easier to get your mind around...
- But, it's hard to build up piece-by-piece in a completely intuitive way...
- **Thus, please have faith that we're going somewhere interesting with all this...**

#### 3.1 Polynomial Operations Over an Algebra

In this section, we'll cover the goal in (14a). The principle means of creating a complex operation from simpler operations is *function composition* (Handout 1), repeated below.

#### (15) (Generalized) Function Composition

Let  $g$  be an  $n$ -ary function, and let  $f_1, \dots, f_n$  be a series of  $n$   $m$ -ary functions. The *composition of  $g$  and  $f_1, \dots, f_n$*  is the  $m$ -ary function defined as follows:

$$g \langle f_1, \dots, f_n \rangle =_{def} \text{the } m\text{-ary function such that for any } m\text{-ary sequence } a_1, \dots, a_m \\ g \langle f_1, \dots, f_n \rangle (\langle a_1, \dots, a_m \rangle) = \\ g( f_1(\langle a_1, \dots, a_m \rangle), \dots, f_n(\langle a_1, \dots, a_m \rangle) )$$

#### Illustration:

Let  $g = \{ \langle \langle x, y \rangle, z \rangle : z = x + y \}$ ,  $f = \{ \langle x, y \rangle : y = x - 1 \}$ ,  $h = \{ \langle x, y \rangle : y = x + 2 \}$

$$\text{Then: } g \langle f, h \rangle (2) = g(f(2), h(2)) = g(1, 4) = 5$$

$$g \langle f, h \rangle = \{ \langle x, y \rangle : y = (x-1) + (x+2) \}$$

In addition to function composition, we'll also make use of the special functions in (16) and (17).

(16) **Identity Functions**

The function  $\text{Id}_{n,m}$  takes as argument an m-tuple  $\alpha$  and returns the nth member of  $\alpha$

a.	$\text{Id}_{1,1}(a)$	=	a			
	$\text{Id}_{1,1}(b)$	=	b			
b.	$\text{Id}_{1,2}(a,b)$	=	a	$\text{Id}_{1,2}(c,d)$	=	c
	$\text{Id}_{2,2}(a,b)$	=	b	$\text{Id}_{2,2}(c,d)$	=	d
c.	$\text{Id}_{1,3}(a,b,c)$	=	a	$\text{Id}_{1,3}(c,d,e)$	=	c
	$\text{Id}_{2,3}(a,b,c)$	=	b	$\text{Id}_{2,3}(c,d,e)$	=	d
	$\text{Id}_{3,3}(a,b,c)$	=	c	$\text{Id}_{3,3}(c,d,e)$	=	e

(17) **Constant Functions**

The function  $C_{\alpha, m}$  takes as argument an m-tuple  $\beta$ , and for any such m-tuple  $\beta$ , returns  $\alpha$

a.	$C_{a,1}(a)$	=	a	$C_{b,1}(a)$	=	b
	$C_{a,1}(b)$	=	a	$C_{b,1}(b)$	=	b
b.	$C_{a,2}(a,b)$	=	a	$C_{b,2}(a,b)$	=	b
	$C_{a,2}(c,d)$	=	a	$C_{b,2}(c,d)$	=	b
c.	$C_{a,3}(a,b,c)$	=	a	$C_{b,3}(a,b,c)$	=	b
	$C_{a,3}(c,d,e)$	=	a	$C_{b,3}(c,d,e)$	=	b

With these ingredients in place, we can introduce the key concept in (18).

(18) **The Polynominal Operations Over an Algebra**

Let  $\mathbf{A} = \langle A, F_\gamma \rangle_{\gamma \in \Gamma}$  be an algebra. The class of polynomial operations over  $\mathbf{A}$  is the smallest class  $K$  such that the following all hold.

- a.  $F_\gamma \in K$  for all  $\gamma \in \Gamma$

Note: The polynomial operations over  $\mathbf{A}$  include all the operations in  $\mathbf{A}$

- b.  $\text{Id}_{n,m} \in K$ , for all  $n, m \in \mathbb{N}$

Note: Every possible identity function is also a polynomial operation over  $\mathbf{A}$ .

- c.  $C_{a,m} \in K$ , for all  $a \in A$  and  $m \in \mathbb{N}$

Note: Every possible constant function (to  $A$ ) is also a polynomial op. over  $\mathbf{A}$ .

- d. If  $G$  is an  $n$ -ary function in  $K$ , and  $F_1, \dots, F_n$  are  $n$  m-ary functions in  $K$ , then  $G \langle F_1, \dots, F_n \rangle \in K$

Note: The polynomial operations over  $\mathbf{A}$  are closed under function composition.

(19) **Remark**

So, in other words,  $F$  is a polynomial operation over  $\mathbf{A}$  ( $= \langle \mathbf{A}, F_\gamma \rangle_{\gamma \in \Gamma}$ ) if any of the following hold:

- a.  $F$  is one of the operations  $\{F_\gamma\}_{\gamma \in \Gamma}$ .
- b.  $F$  is an identity function.
- c.  $F$  is a constant function (to  $\mathbf{A}$ ).
- d. You can obtain  $F$  via iterated function composition from either (a), (b), or (c).

*We'll now illustrate the key concept in (18) by looking at some polynomial operations over the syntactic algebra for Politics-NoQ,  $\langle \mathbf{A}, F_\gamma \rangle_{\gamma \in \{\text{Concat, Not, And}\}}$ .*

(20)  **$C_{(\text{smokes' barack')}, 1}$**

This function takes any expression in  $\mathbf{A}$  and returns the expression **(smokes' barack')**

$$C_{(\text{smokes' barack')}, 1} (\text{(loves' michelle')}) = \text{(smokes' barack')} \\ C_{(\text{smokes' barack')}, 1} (\text{mitt'}) = \text{(smokes' barack')}$$

(21)  **$\mathbf{Id}_{2,3}$**  This function takes any triple in  $\mathbf{A}$  and returns the second member.

(22)  **$\mathbf{Id}_{1,2}$**  This function takes any pair in  $\mathbf{A}$  and returns the first member.

(23)  **$\mathbf{Id}_{2,2}$**  This function takes any pair in  $\mathbf{A}$  and returns the second member.

(24)  **$F_{Not} < F_{Not} >$**

This function takes any expression in  $\mathbf{A}$  and returns its double negation.

$$F_{Not} < F_{Not} > (\text{(smokes' barack')}) = \\ F_{Not}(F_{Not}(\text{(smokes' barack')})) = \\ \sim\sim(\text{smokes' barack'})$$

(25)  **$F_{And} < C_{(\text{smokes' barack')}, 1}, \mathbf{Id}_{1,1} >$**

This function takes any expression  $\alpha$  in  $\mathbf{A}$  and returns the conjunction of **(smokes' barack')** with  $\alpha$ .

$$F_{And} < C_{(\text{smokes' barack')}, 1}, \mathbf{Id}_{1,1} > (\text{(smokes' mitt')}) = \\ F_{And}(C_{(\text{smokes' barack')}, 1}((\text{smokes' mitt'})), \mathbf{Id}_{1,1}((\text{smokes' mitt'}))) = \\ F_{And}(\text{(smokes' barack')}, (\text{smokes' mitt'})) = \\ (\text{(smokes' barack') \& (smokes' mitt')})$$

(26)  $F_{Not} < F_{And} < \text{Id}_{1,2}, F_{Not} < \text{Id}_{2,2} >>>$

This function takes any pair of expressions  $\alpha, \beta$  in A and returns  $\sim(\alpha \& \sim\beta)$ .

$$\begin{aligned}
 F_{Not} < F_{And} < \text{Id}_{1,2}, F_{Not} < \text{Id}_{2,2} >>> (\text{(smokes' barack')}, \text{(smokes' mitt')}) &= \\
 F_{Not}( F_{And}( \text{Id}_{1,2}(\text{(smokes' barack')}, \text{(smokes' mitt')}) ) \\
 F_{Not}( \text{Id}_{2,2}(\text{(smokes' barack')}, \text{(smokes' mitt')}) ) \dots ) &= \\
 F_{Not}( F_{And}( \text{(smokes' barack')}, F_{Not}( \text{(smokes' mitt')}) \dots ) &= \\
 F_{Not}( F_{And}( \text{(smokes' barack')}, \sim(\text{smokes' mitt'})) \dots ) &= \\
 F_{Not}( ( \text{(smokes' barack')} \& \sim(\text{smokes' mitt'})) ) &= \\
 \sim( \text{(smokes' barack')} \& \sim(\text{smokes' mitt'})) &
 \end{aligned}$$

(27) **Key Observation**

Recall that the formulae  $(\varphi \rightarrow \psi)$  and  $\sim(\varphi \& \sim\psi)$  are logically equivalent.

- Consequently, we are viewing  $(\varphi \rightarrow \psi)$  as a special ‘abbreviation’ for  $\sim(\varphi \& \sim\psi)$
- Thus, the operation  $F_{Not} < F_{And} < \text{Id}_{1,2}, F_{Not} < \text{Id}_{2,2} >>>$  would seem to be a good ‘translational correspondent’ of  $K_{If}$  in Disambiguated Mini-English**

$$h(K_{If}(\alpha, \beta)) = F_{Not} < F_{And} < \text{Id}_{1,2}, F_{Not} < \text{Id}_{2,2} >>>(h(\alpha), h(\beta)) = \sim(h(\alpha) \& \sim h(\beta)) = (h(\alpha) \rightarrow h(\beta))$$

(28)  $F_{Concat} < \text{Id}_{2,2}, \text{Id}_{1,2} >$

This function takes any pair of expressions  $\alpha, \beta$  in A and returns  $(\beta \alpha)$

$$\begin{aligned}
 F_{Concat} < \text{Id}_{2,2}, \text{Id}_{1,2} > (\text{barack'}, \text{smokes'}) &= \\
 F_{Concat}( \text{Id}_{2,2}(\text{barack'}, \text{smokes'}), \text{Id}_{1,2}(\text{barack'}, \text{smokes'})) &= \\
 F_{Concat}( \text{smokes'}, \text{barack'}) &= \\
 (\text{smokes'} \text{ barack'}) &
 \end{aligned}$$

(29) **Key Observation**

It seems like  $F_{Concat} < \text{Id}_{2,2}, \text{Id}_{1,2} >$  would be a good ‘translational correspondent’ of  $K_{Merge-S}$  in Disambiguated Mini-English.

$$\begin{aligned}
 h(K_{Merge-S}(\text{Barack}, \text{smokes})) = F_{Concat} < \text{Id}_{2,2}, \text{Id}_{1,2} > (h(\text{Barack}), h(\text{smokes})) &= \\
 F_{Concat} < \text{Id}_{2,2}, \text{Id}_{1,2} > (\text{barack'}, \text{smokes'}) &= (\text{smokes'} \text{ barack'})
 \end{aligned}$$

### (30) Summary Observation

- If we look to the **polynomial operations over** the syntactic algebra for Politics-NoQ,  $\langle A, F_\gamma \rangle_{\gamma \in \{\text{Concat, Not, And}\}}$ , we will find syntactic operations over A that could viably correspond to the syntactic operations  $K_{\text{If}}$  and  $K_{\text{Merge-S}}$  over E
- *How, though, does this help us in our quest for a homomorphism from E to A?...*

### (31) Polynomial Operations and Algebras

a. Key Fact:

Let  $A$  be an algebra  $\langle A, F_\gamma \rangle_{\gamma \in \Gamma}$ . If the set  $\{ H_\gamma \}_{\gamma' \in \Gamma'}$  consists of polynomial operations over  $A$ , then  $A$  is closed under  $\{ H_\gamma \}_{\gamma' \in \Gamma'}$ .

(proof left as an exercise to the student)

b. Key Consequence:

Let  $A$  be an algebra  $\langle A, F_\gamma \rangle_{\gamma \in \Gamma}$ , and let  $\{ H_\gamma \}_{\gamma' \in \Gamma'}$  consist of polynomial operations over  $A$ . The structure  $\langle A, H_\gamma \rangle_{\gamma' \in \Gamma'}$ , is an algebra.

Thus, given (31b), if  $\langle A, F_\gamma \rangle_{\gamma \in \{\text{Concat, Not, And}\}}$  is the syntactic algebra for Politics-NoQ, then the following is also an algebra:  $\langle A, H_\gamma \rangle_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$ , where

- $H_{\text{Not}}$  and  $H_{\text{And}} = F_{\text{Not}}$  and  $F_{\text{And}}$ , respectively
- $H_{\text{Merge-IV}} = F_{\text{Concat}}$
- $H_{\text{If}} = F_{\text{Not}} \langle F_{\text{And}} \langle \text{Id}_{1,2}, F_{\text{Not}} \langle \text{Id}_{2,2} \rangle \rangle \rangle$
- $H_{\text{Merge-S}} = F_{\text{Concat}} \langle \text{Id}_{2,2}, \text{Id}_{1,2} \rangle$

**And, it seems like it might be possible to have a homomorphism from the syntactic algebra for DME  $\langle E, K_\gamma \rangle_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$  to  $\langle A, H_\gamma \rangle_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$**

*Before we can use all of this to lay out a theory of homomorphic translation between languages, we also need to introduce a way of constructing complex syntactic rules (14b)...*

### 3.2 The Derived Syntactic Rules of a Language L

#### (32) Background Motivation

- Intuitively, a translation from one language  $L$  to another language  $L'$  should always map the well-formed expressions of  $L$  to well-formed expressions of  $L'$
- As we'll soon see, one way of ensuring this appeals to the notion of a 'derived syntactic rule', defined in this section.
- **For reasons that will also be clear shortly, this definition is going to closely mirror the definition for the polynomial operations over an algebra**

#### (33) Derived Syntactic Rules of a Language

Let  $\mathbf{L}$  be a language  $\langle\langle A, F_\gamma, X_\delta, S, \delta_0 \rangle\rangle_{\gamma \in \Gamma, \delta \in \Delta, R}$ . The derived syntactic rules of  $\mathbf{L}$  is the smallest class  $K$  such that the following hold:

a.  $S \subseteq K$

Note: Thus all the syntactic rules of  $\mathbf{L}$  are also 'derived syntactic rules'

b. For all  $n, m \in \mathbb{N}$  such that  $n \leq m$ , if  $\langle \delta_1, \dots, \delta_n, \dots, \delta_m \rangle$  is an  $m$ -tuple of elements from  $\Delta$ , then  $\langle \text{Id}_{n,m}, \langle \delta_1, \dots, \delta_n, \dots, \delta_m \rangle, \delta_n \rangle \in K$

Note:

This means that all the logically possible syntactic rules of the form below are also 'derived syntactic rules':

$$\langle \text{Id}_{2,4}, \langle e, t, \langle e, \langle e, t \rangle \rangle, \langle e, t \rangle \rangle, t \rangle$$

'The result of applying  $\text{Id}_{2,4}$  to a quadruple consisting of an expression of type  $e$ , one of type  $t$ , one type  $\langle e, \langle e, t \rangle \rangle$  and one of type  $\langle e, t \rangle$  is an expression of type  $t$ '

c. For all  $n \in \mathbb{N}$ , if  $a \in C_\delta$ , and  $\langle \delta_1, \dots, \delta_n \rangle$  is an  $n$ -tuple of elements from  $\Delta$ , then the triple  $\langle C_{a,n}, \langle \delta_1, \dots, \delta_n \rangle, \delta \rangle \in K$ .

Note:

This means that all the logically possible syntactic rules of the form below are also 'derived syntactic rules'.

$$\langle C_{(\text{smokes' barack}'), 2}, \langle \langle e, t \rangle, \langle e, t \rangle \rangle, t \rangle$$

'The result of applying  $C_{(\text{smokes' barack}'), 2}$  to a pair consisting of an expression of type  $\langle e, t \rangle$  and an expression of type  $\langle e, t \rangle$  is an expression of type  $t$ '

*The fourth and final condition on the derived syntactic rules basically amounts to them being closed under 'composition'... it is rather complex to state formally, however...*

- d. If  $\langle F, \langle \delta_1, \dots, \delta_n \rangle, \delta \rangle \in K$ ,  $F$  is an  $n$ -ary operation, and each of  $G_1, \dots, G_n$  are an  $m$ -ary operation such that  $\langle G_j, \langle \delta'_1, \dots, \delta'_m \rangle, \delta_j \rangle \in K$ , then the following is also a member of  $K$ :

$$\langle F \langle G_1, \dots, G_n \rangle, \langle \delta'_1, \dots, \delta'_m \rangle, \delta \rangle$$

Note: To get a sense of how this ‘composition’ operation on rules works, consider that (i) and (ii) are rules in our language Politics-NoQ.

- (i)  $\langle F_{\text{Not}}, \langle t \rangle, t \rangle$
- (ii)  $\langle F_{\text{Concat}}, \langle \langle e, t \rangle, e \rangle, t \rangle$

Thus, definition (33) would entail that the following is a ‘derived rule’ of Politics-NoQ.

- (iii)  $\langle F_{\text{Not}} \langle F_{\text{Concat}} \rangle, \langle \langle e, t \rangle, e \rangle, t \rangle$   
‘The result of applying  $F_{\text{Not}} \langle F_{\text{Concat}} \rangle$  to a pair consisting of an expression of type  $\langle e, t \rangle$  and an expression of type  $e$  is an expression of type  $t$ .’

Note, too, that this derived rule would intuitively ‘be true of’ for Politics-NoQ.

- This raises the following key generalization...

#### (34) Derived Syntactic Rules and the Syntactic Categories of a Language

Let  $L$  be a language  $\langle \langle A, F_\gamma, X_\delta, S, \delta_0 \rangle_\gamma \in \Gamma, \delta \in \Delta, R \rangle$ , and let  $\langle H, \langle \delta_1, \dots, \delta_n \rangle, \delta \rangle$  be a derived syntactic rule of  $L$ .

- a. Claim: If  $\varphi_1, \dots, \varphi_n$  are such that each  $\varphi_i \in C_{\delta_i}$ , then  $H(\varphi_1, \dots, \varphi_n) \in C_\delta$   
(That is, the derived syntactic rules will only ever generate ‘meaningful expressions’ of a language.)
- b. Proof: (left as an exercise to the student)

## 4. The Concept of a Translation Base

In the previous section, we developed the tools below. We also developed them so that they mirror one another.

#### (35) Polynomial Operations (Over an Algebra)

A way of taking algebra  $\langle A, F_\gamma \rangle_{\gamma \in \Gamma}$  and creating new complex operations  $\{H_\gamma\}_{\gamma' \in \Gamma'}$  from the operations  $\{F_\gamma\}_{\gamma \in \Gamma}$  such that  $\langle A, H_{\gamma'} \rangle_{\gamma' \in \Gamma'}$  is also an algebra.

(36) **Derived Syntactic Rules**

A way of taking a language  $\langle A, F_\gamma, X_\delta, S, \delta_0 \rangle_{\gamma \in \Gamma, \delta \in \Delta}$  and creating new complex syntactic rules  $S'$  from the polynomial operations over  $\langle A, F_\gamma \rangle_{\gamma \in \Gamma}$ .

We're now going to use these tools to construct Montague's general theory of translation...

To do this, let's first consider some ideal properties of a translation function  $h$  from one language  $\langle A, F_\gamma, X_\delta, S, \delta_0 \rangle_{\gamma \in \Gamma, \delta \in \Delta}$  to another language  $\langle A', F'_{\gamma'}, X'_{\delta'}, S', \delta'_0 \rangle_{\gamma' \in \Gamma', \delta' \in \Delta'}$

(37) **Correspondence Between the Syntactic Categories**

In Montague's theory of translation, there must be a consistent mapping from the syntactic categories of  $L$  to the syntactic categories of  $L'$ .

- That is, if  $\delta \in \Delta$ , then there must be a corresponding  $\delta' \in \Delta'$  such that if  $\varphi \in C_\delta$ , then  $h(\varphi) \in C_{\delta'}$
- For example, thinking of our languages DME and Politics-NoQ, such a mapping of the categories would be as follows:

$$\begin{aligned} NP &\rightarrow e \\ TV &\rightarrow \langle e, \langle e, t \rangle \rangle \\ IV &\rightarrow \langle et \rangle \\ S &\rightarrow t \end{aligned}$$

- The reason why such a mapping is needed is ultimately tied to Montague's (final) definition of a 'Fregean Interpretation' ... Just go with it for now...

**Consequence:** A translation from  $L$  to  $L'$  will need to specify a function  $g$  from  $\Delta$  to  $\Delta'$

(38) **Polynomial Operations**

If we want the translation  $h: A \rightarrow A'$  from  $L$  to  $L'$  to be a homomorphism, then we're going to need to find some operations  $F'$  to correspond with the syntactic operations of  $L$ .

- We've already seen that in the general case the basic operations  $\{ F'_{\gamma'} \}_{\gamma' \in \Gamma'}$  of  $L'$  are not going to be sufficient.
- We've also already seen that the polynomial operations over the syntactic algebra for  $L', \langle A', F'_{\gamma'} \rangle_{\gamma' \in \Gamma'}$  can supply us with such operations.

**Consequence:**

A translation from  $L$  to  $L'$  must identify some polynomial operations  $\{ H_\gamma \}_{\gamma \in \Gamma}$  over the syntactic algebra for  $L', \langle A', F'_{\gamma'} \rangle_{\gamma' \in \Gamma'}$

(39) **Derived Syntactic Rules**

A translation  $h$  from  $L$  to  $L'$  should always map the ‘meaningful expressions’ of  $L$  to meaningful expressions in  $L'$ .

- Now, recall that we’re going to want  $h$  to be a homomorphism, where every syntactic operation  $F_\gamma$  in  $L$  corresponds with some polynomial operation  $H_\gamma$  over the syntactic algebra of  $L'$ .

$$h(F_\gamma(\alpha_1, \dots, \alpha_n)) = H_\gamma(h(\alpha_1), \dots, h(\alpha_n))$$

- Consequently, we will want it to be that if  $F_\gamma(\alpha_1, \dots, \alpha_n)$  is a meaningful expression of  $L$ , then  $H_\gamma(h(\alpha_1), \dots, h(\alpha_n))$  is a meaningful expression of  $L'$  too.

**Consequence:**

Under a translation  $h$  from  $L$  to  $L'$ , if  $F_\gamma$  in  $L$  corresponds with  $H_\gamma$  (a polynomial operation over  $L'$ ), then if (a) is a **syntactic rule** of  $L$ , then (b) is a **derived syntactic rule** of  $L'$

- a.  $\langle F_\gamma, \langle \delta_1, \dots, \delta_n \rangle, \delta \rangle$
- b.  $\langle H_\gamma, \langle g(\delta_1), \dots, g(\delta_n) \rangle, g(\delta) \rangle$

- To see how the condition above works, recall the general result in (34): if the tuple  $\langle H_\gamma, \langle g(\delta_1), \dots, g(\delta_n) \rangle, g(\delta) \rangle$  is a derived syntactic rule, and  $\varphi_1, \dots, \varphi_n$  are such that each  $\varphi_i \in C_{g(\delta_i)}$ , then  $H(\varphi_1, \dots, \varphi_n) \in C_{g(\delta)}$ 
  - Now, suppose that  $\alpha$  is a meaningful expression of  $L$ , and  $F_\gamma(\alpha_1, \dots, \alpha_n) = \alpha$ , where  $\alpha_1 \in \delta_1, \dots, \alpha_n \in \delta_n$
  - Thus, the translation  $h(\alpha) = h(F_\gamma(\alpha_1, \dots, \alpha_n)) = H_\gamma(h(\alpha_1), \dots, h(\alpha_n))$
  - Now, given our category correspondence (37), it follows that  $h(\alpha_1) \in C_{g(\delta_1)}, \dots, h(\alpha_n) \in C_{g(\delta_n)}$
  - Therefore, from our general result in (34) – and the fact that (b) is a derived rule of  $L'$  – it follows that  $H_\gamma(h(\alpha_1), \dots, h(\alpha_n)) \in C_{g(\delta)}$
  - Thus, we have it that  $h(\alpha)$  is also a meaningful expression of  $L'$ !**

*With all of these ingredients on the table, we can now provide Montague’s general definition of a ‘translation base’...*

(40) **Translation Base from Language L to Language L'**

Let  $\mathbf{L}$  be a language  $\langle L, R \rangle$  and  $\mathbf{L}'$  be a language  $\langle L', R' \rangle$ , where  $L$  is the disambiguated language  $\langle A, F_\gamma, X_\delta, S, \delta_0 \rangle_{\gamma \in \Gamma, \delta \in \Delta}$  and  $L' = \langle A', F'_{\gamma'}, X'_{\delta'}, S', \delta'_0 \rangle_{\gamma' \in \Gamma', \delta' \in \Delta'}$

A **translation base from L to L'** is a structure  $\langle g, H_\gamma, j \rangle_{\gamma \in \Gamma}$  such that:

a.  $g$  is a function from  $\Delta$  to  $\Delta'$  (37)

b. For all  $\gamma \in \Gamma$ ,  $H_\gamma$  is a polynomial operation over the algebra  $\langle A', F'_{\gamma'} \rangle_{\gamma' \in \Gamma'}$  sharing the same arity as  $F_\gamma$  (38)

c. If  $\langle F_\gamma, \langle \delta_1, \dots, \delta_n \rangle, \delta \rangle \in S$ , then the following is a derived syntactic rule for  $\mathbf{L}'$ :  
 $\langle H_\gamma, \langle g(\delta_1), \dots, g(\delta_n) \rangle, g(\delta) \rangle$  (39)

d.  $j$  is a function whose domain is  $\cup_{\delta \in \Delta} X_\delta$ , and whenever  $\varphi \in X_\delta$ ,  $j(\varphi) \in C'_{g(\delta)}$ .

Note:  $j$  is a function that maps the basic expressions of  $\mathbf{L}$  of category  $\delta$  to some meaningful expressions of  $\mathbf{L}'$  of the corresponding category  $g(\delta)$ .

*With the notion of a translation base, we can construct the following definition of a translation function...*

(41) **Translation Function from Language L to Language L'**

Let  $\mathbf{L}$  be a language  $\langle L, R \rangle$  and  $\mathbf{L}'$  be a language  $\langle L', R' \rangle$ , where  $L$  is the disambiguated language  $\langle A, F_\gamma, X_\delta, S, \delta_0 \rangle_{\gamma \in \Gamma, \delta \in \Delta}$  and  $L' = \langle A', F'_{\gamma'}, X'_{\delta'}, S', \delta'_0 \rangle_{\gamma' \in \Gamma', \delta' \in \Delta'}$   
Let  $\mathbf{T}$  be a translation base  $\langle g, H_\gamma, j \rangle_{\gamma \in \Gamma}$  from  $\mathbf{L}$  to  $\mathbf{L}'$

The **translation function determined by T** is the unique homomorphism  $k$  from the algebra  $\langle A, F_\gamma \rangle_{\gamma \in \Gamma}$  to the algebra  $\langle A', H_\gamma \rangle_{\gamma \in \Gamma}$  such that  $j \subseteq k$ .

(42) **Remarks**

- That any translation base  $\mathbf{T}$  determines such a homomorphism  $k$  is essentially guaranteed by the conditions we placed on the polynomial operations  $\{ H_\gamma \}_{\gamma \in \Gamma}$
- If  $k$  is a translation function from  $\mathbf{L}$  to  $\mathbf{L}'$ , then  $k$  is not *necessarily* a homomorphism from the syntactic algebra of  $\mathbf{L}$  to the syntactic algebra of  $\mathbf{L}'$ 
  - Rather, it's a homomorphism to an algebra we define on the basis of  $\mathbf{L}'$
  - This allows translation to be a homomorphism even if two language algebras are not themselves homomorphic!

(43) **Definition of Translation**

Let  $\mathbf{L}$  be a language  $\langle L, R \rangle$  and  $\mathbf{L}'$  be a language  $\langle L', R' \rangle$ , where  $L$  is the disambiguated language  $\langle A, F_\gamma, X_\delta, S, \delta_0 \rangle_{\gamma \in \Gamma, \delta \in \Delta}$  and  $L' = \langle A', F'_\gamma, X'_\delta, S', \delta'_0 \rangle_{\gamma' \in \Gamma', \delta' \in \Delta'}$ . Let  $\mathbf{T}$  be a translation base from  $\mathbf{L}$  to  $\mathbf{L}'$ , and let  $k$  be the translation function determined by  $\mathbf{T}$ .

If  $\alpha$  is an expression of  $\mathbf{L}'$  and  $\beta$  is an expression of  $\mathbf{L}$ , then  $\alpha$  is a **translation of**  $\beta$  if there are  $\alpha' \in A'$  and  $\beta' \in A$  such that:

- (i)  $\alpha' R' \alpha$  and  $\beta' R \beta$
  - (ii)  $\alpha'$  is a meaningful expression of  $L'$  and  $\beta'$  is a meaningful expression of  $L$
  - (iii)  $k(\beta') = \alpha'$
- 

**5. Translating from Mini-English to Politics-NoQ**

To round out these notes, we'll use all these tools to spell out a translation base and (homomorphic) translation function from Mini-English to Politics-NoQ

(44) **Mini-English**

Mini-English is the structure  $\langle \langle E, K_\gamma, X_\delta, S_E, S \rangle_\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}, \delta \in \Delta, R \rangle$ , where

- a. The structure  $\langle E, K_\gamma, X_\delta, S_E, S \rangle_\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}, \delta \in \Delta$  is Disambiguated Mini-English, as defined in (10).
- b.  $R$  is a function which takes as input a tree  $T$  in  $E$ , and returns as output the first member of the root node of  $T$ .

(45) **Politics-NoQ**

Politics-NoQ is the structure  $\langle \langle A, F_\gamma, X_\tau, S, t \rangle_\gamma \in \{\text{Concat, Not, And}\}, \tau \in T, R' \rangle$ , where:

- a. The structure  $\langle A, F_\gamma, X_\tau, S, t \rangle_\gamma \in \{\text{Concat, Not, And}\}, \tau \in T$  is the disambiguated language Politics-NoQ, as defined previously.
- b.  $R$  is the identity function.

We'll now lay out each of the three main ingredients for a translation base from Mini-English to Politics-NoQ...

(46) **Correspondence Between the Syntactic Categories**

Given our discussion in (37), let us define the function  $g: \Delta \rightarrow T$  as follows:

$$g(NP) = e \quad g(TV) = \langle e, \langle e, t \rangle \rangle \quad g(IV) = \langle e, t \rangle \quad g(S) = t$$

(47) **The Polynomial Operations**

Given our discussion below (31), we will want to build our translation base from the following polynomial operations over the algebra  $\langle A, F_\gamma \rangle_{\gamma \in \{\text{Concat}, \text{Not}, \text{And}\}}$ .

- a.  $H_{\text{Not}}$  and  $H_{\text{And}}$  =  $F_{\text{Not}}$  and  $F_{\text{And}}$ , respectively
- b.  $H_{\text{Merge-IV}}$  =  $F_{\text{Concat}}$
- c.  $H_{\text{If}}$  =  $F_{\text{Not}} \langle F_{\text{And}} \langle \text{Id}_{1,2}, F_{\text{Not}} \langle \text{Id}_{2,2} \rangle \rangle \rangle$
- d.  $H_{\text{Merge-S}}$  =  $F_{\text{Concat}} \langle \text{Id}_{2,2}, \text{Id}_{1,2} \rangle$

(48) **Crucial Step We Will Usually Leave Implicit**

If we build our translation base on the polynomial operations above, then condition (39)/(40c) – combined with our category correspondence in (46) – requires the following:

- a.  $\langle H_{\text{Not}}, \langle t \rangle, t \rangle$  is a derived rule of Politics-NoQ
- b.  $\langle H_{\text{And}}, \langle t, t \rangle, t \rangle$  is a derived rule of Politics-NoQ
- c.  $\langle H_{\text{Merge-IV}}, \langle \langle e, \langle e, t \rangle \rangle, e \rangle, \langle e, t \rangle \rangle$  is a derived rule of Politics-NoQ
- d.  $\langle H_{\text{If}}, \langle t, t \rangle, t \rangle$  is a derived rule of Politics-NoQ
- e.  $\langle H_{\text{Merge-S}}, \langle e, \langle et \rangle \rangle, t \rangle$  is a derived rule of Politics-NoQ

*Showing that (48a-e) hold will be left as an exercise for the student. Note that (48a,b,c) are trivial; only (48d,e) require some calculating out...*

(49) **The (Lexical Translation) Function  $j$**

Given the category correspondence in (46) – and what we obviously want to achieve – let us define the function  $j$  as follows:

- a.  $j(\langle \text{Barack}, \emptyset \rangle) = \text{barack}'$
- b.  $j(\langle \text{Michelle}, \emptyset \rangle) = \text{michelle}'$
- c.  $j(\langle \text{Mitt}, \emptyset \rangle) = \text{mitt}'$
- d.  $j(\langle \text{smokes}, \emptyset \rangle) = \text{smokes}'$
- e.  $j(\langle \text{loves}, \emptyset \rangle) = \text{loves}'$

(50) **Putting It All Together: The Translation Base from Mini-English to Politics-NoQ**

Let  $\mathbf{T}$  be the structure  $\langle g, H_\gamma, j \rangle_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$ , where  $g$ ,  $H_\gamma$ , and  $j$  are as defined in (46)-(49).  $\mathbf{T}$  is a translation base from Mini-English to Politics-NoQ.

*We can now use the translation function  $k$  determined by  $\mathbf{T}$  to homomorphically map expressions of mini-English to expressions of Politics-NoQ.*

(51) **Translating from Mini-English to Politics-NoQ**

Let  $k$  be the translation function determined by  $\mathbf{T}$ , as defined in (50). Let  $T$  be the tree in (10) above.

- a.  $k(T) =$  (by definition of DME)
- b.  $k(K_{\text{Merge-S}}(\langle Barack, \emptyset \rangle, K_{\text{Merge-IV}}(\langle loves \emptyset \rangle, \langle Michelle, \emptyset \rangle))) =$  (by homomorphism property of  $k$ )
- c.  $H_{\text{Merge-S}}(k(\langle Barack, \emptyset \rangle), k(K_{\text{Merge-IV}}(\langle loves \emptyset \rangle, \langle Michelle, \emptyset \rangle))) =$  (by homomorphism property of  $k$ )
- d.  $H_{\text{Merge-S}}(k(\langle Barack, \emptyset \rangle), H_{\text{Merge-IV}}(k(\langle loves \emptyset \rangle), k(\langle Michelle, \emptyset \rangle))) =$  (by definition of  $k$  and  $j$ )
- e.  $H_{\text{Merge-S}}(j(\langle Barack, \emptyset \rangle), H_{\text{Merge-IV}}(j(\langle loves \emptyset \rangle), j(\langle Michelle, \emptyset \rangle))) =$  (by definition of  $j$ )
- f.  $H_{\text{Merge-S}}(\mathbf{barack}', H_{\text{Merge-IV}}(\mathbf{loves}', \mathbf{michelle}')) =$  (by definition of  $H_{\text{Merge-IV}}$ )
- g.  $H_{\text{Merge-S}}(\mathbf{barack}', (\mathbf{loves}' \mathbf{michelle}')) =$  (by definition of  $H_{\text{Merge-S}}$ )
- h.  $((\mathbf{loves}' \mathbf{michelle}') \mathbf{barack}')$

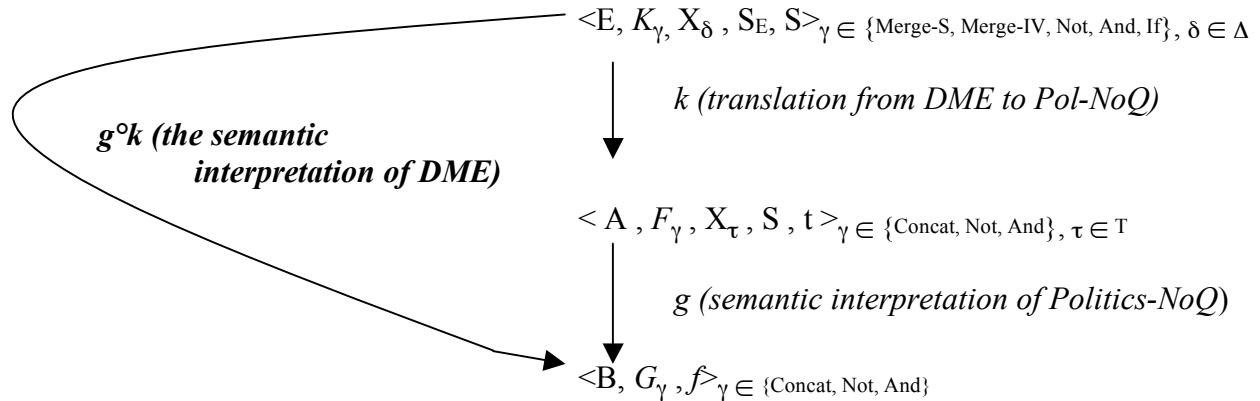
(52) **Remark**

Under the translation function  $k$  determined by  $\mathbf{T}$ , and given the definition in (43), it follows that there are *two* different formulae in Politics-NoQ that are translations of the Mini-English sentence *It is not the case that Barack smokes and Mitt smokes*.

(exercise for the student!)

(53) **What We Wanted**

We wanted to develop a way of homomorphically mapping expressions of DME to expressions of Politics-NoQ (so that we can ultimately get a semantics for English)



(54) **What We Have Now**

$$\begin{array}{ccc}
 <E, K_\gamma>_\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\} & & <A, F_\gamma>_\gamma \in \{\text{Concat, Not, And}\} \\
 \downarrow k & & \downarrow g \\
 <A, H_\gamma>_\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\} & & <B, G_\gamma>_\gamma \in \{\text{Concat, Not, And}\}
 \end{array}$$

- We have a way of homomorphically mapping expressions of Mini-English to expressions of Politics-NoQ.
- **However, our translation homomorphism doesn't hold between the syntactic algebra of Mini-English and the syntactic algebra of Politics-NoQ.**
  - Rather, it holds between the syntactic algebra of Mini-English and *another* syntactic algebra that we construct on the basis of  $<A, F_\gamma>_\gamma \in \{\text{Concat, Not, And}\}$
- **But, our interpretation homomorphism holds between the syntactic algebra of Politics-NoQ and the interpretation  $<B, G_\gamma, f>_\gamma \in \{\text{Concat, Not, And}\}$** 
  - Our interpretation function  $g$  is not a homomorphism from our derived syntactic algebra  $<A, H_\gamma>_\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}$
- **Therefore, the composition of  $k$  and  $g$  is *not* a homomorphism from our syntactic algebra for Mini-English to the interpretation  $<B, G_\gamma, f>_\gamma \in \{\text{Concat, Not, And}\}$** 
  - *So how do we get what we want, a homomorphism from the syntactic algebra of Mini-English to an interpretation structure?....*

Tune in next time for Part 3 of ‘Montague’s Theory of Translation’...

## Montague's Theory of Translation: Translation Bases and Indirect Interpretations<sup>1</sup>

### 1. From Translation Functions to Interpretations

#### (1) What We Currently Have

##### a. Mini-English (and Disambiguated-Mini-English):

Mini-English is  $\langle\langle E, K_\gamma, X_\delta, S_E, S \rangle\rangle_\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}, \delta \in \Delta, R \rangle$

(i)  $\langle E, K_\gamma, X_\delta, S_E, S \rangle_\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}, \delta \in \Delta$  is Disambiguated Mini-English

(ii) R maps trees in E to the first member of their root node.

##### b. Politics-NoQ

Politics-NoQ is the language  $\langle\langle A, F_\gamma, X_\tau, S, t \rangle\rangle_\gamma \in \{\text{Concat, Not, And}\}, \tau \in T, Id \rangle$

(i)  $\langle A, F_\gamma, X_\tau, S, t \rangle_\gamma \in \{\text{Concat, Not, And}\}, \tau \in T$  is the disambiguated language Politics-NoQ.

(ii) Id is the identity function.

##### c. Interpretation for Politics-NoQ

Let the set S = { Michelle, Barack, Mitt }. Let  $\mathbf{B} = \langle B, G_\gamma, f \rangle_\gamma \in \{\text{Concat, Not, And}\}$  be the Fregean interpretation based on S, such that f is as defined before.

##### d. Translation Base from Mini-English to Politics-NoQ

Let be  $\mathbf{T}$  is the translation base  $\langle g, H_\gamma, j \rangle_\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}$ , where g,  $H_\gamma$ , and j are as defined before.

Where k is the translation function determined by  $\mathbf{T}$  and h is the meaning assignment determined by  $\mathbf{B}$ , we have the following homomorphisms.

$$\begin{array}{ccc} \langle E, K_\gamma \rangle_\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\} & & \\ \downarrow k & & \\ \langle A, H_\gamma \rangle_\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\} & & \end{array}$$

$$\begin{array}{ccc} \langle A, F_\gamma \rangle_\gamma \in \{\text{Concat, Not, And}\} & & \\ \downarrow h & & \\ \langle B, G_\gamma \rangle_\gamma \in \{\text{Concat, Not, And}\} & & \end{array}$$

#### (2) Key Issue

- What we want is a homomorphism that maps E to B.
- **But since the operation indices are different in  $\langle A, H_\gamma \rangle_\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}$  and  $\langle A, F_\gamma \rangle_\gamma \in \{\text{Concat, Not, And}\}$  we don't have such a homomorphism yet...**

---

<sup>1</sup> These notes are based upon material in the following readings: Halvorsen & Ladusaw (1979), Dowty *et al.* (1981) Chapter 8, and Thomason (1974) Chapter 7 (Montague's "Universal Grammar").

(3) **The Key Theorem Relating Translation to Interpretation**

If  $\langle A, F_\gamma \rangle_{\gamma \in \Gamma}$  is an algebra,  $h$  is a homomorphism from  $\langle A, F_\gamma \rangle_{\gamma \in \Gamma}$  to some algebra  $\langle B, G_\gamma \rangle_{\gamma \in \Gamma}$ , and for each  $\gamma \in \Pi$ ,  $H_\gamma$  is a polynomial operation over  $\langle A, F_\gamma \rangle_{\gamma \in \Gamma}$ , then there is exactly one algebra  $\langle B, G_\gamma \rangle_{\gamma \in \Pi}$  such that  $h$  is a homomorphism from  $\langle A, H_\gamma \rangle_{\gamma \in \Pi}$  to  $\langle B, G_\gamma \rangle_{\gamma \in \Pi}$

(Montague 1974: 225)

(4) **Restatement of the Key Theorem**

Suppose the following conditions all hold:

- a. There are two algebras  $A = \langle A, F_\gamma \rangle_{\gamma \in \Gamma}$  and  $B = \langle B, G_\gamma \rangle_{\gamma \in \Gamma}$
- b. There is a homomorphism  $h$  from  $A$  to  $B$
- c. There is a an algebra  $A' = \langle A, H_\gamma \rangle_{\gamma \in \Pi}$  where  $\{H_\gamma\}_{\gamma \in \Pi}$  are all polynomial operations over  $\langle A, F_\gamma \rangle_{\gamma \in \Gamma}$

There is therefore one (exactly one) algebra  $B' = \langle B, G_\gamma \rangle_{\gamma \in \Pi}$  such that  $h$  is also a homomorphism from  $A'$  to  $B'$ .

(5) **The Importance of the Key Theorem**

- Conditions (4a-c) are exactly what we have in (1)!
- a.  $\langle A, F_\gamma \rangle_{\gamma \in \{\text{Concat, Not, And}\}}$  and  $\langle B, G_\gamma \rangle_{\gamma \in \{\text{Concat, Not, And}\}}$  are algebras.
- b. Meaning assignment  $h$  is a homomorphism from  $A \langle A, F_\gamma \rangle_{\gamma \in \{\text{Concat, Not, And}\}}$  to  $B \langle B, G_\gamma \rangle_{\gamma \in \{\text{Concat, Not, And}\}}$
- c.  $\langle A, H_\gamma \rangle_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$  is an algebra where the operations are all polynomial operations over  $\langle A, F_\gamma \rangle_{\gamma \in \{\text{Concat, Not, And}\}}$
- Therefore, the key theorem in (3)/(4) guarantees us the following:  
**There is an algebra  $\langle B, G_\gamma \rangle_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$  such that  $h$  is also a homomorphism to it from  $\langle A, H_\gamma \rangle_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$**

$$\langle E, K_\gamma \rangle_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$$

$k$

$$\langle A, H_\gamma \rangle_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$$

$h$

$$\langle B, G_\gamma \rangle_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$$

$$\langle A, F_\gamma \rangle_{\gamma \in \{\text{Concat, Not, And}\}}$$

$h$

$$\langle B, G_\gamma \rangle_{\gamma \in \{\text{Concat, Not, And}\}}$$

## (6) Remarks

- Note that in the algebra  $\langle B, G_\gamma \rangle_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$  the set B is the set of meanings in our interpretation of Politics-NoQ
- Although Montague doesn't say it, the operations  $\{G_\gamma\}_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$  are all polynomial operations over  $\langle B, G_\gamma \rangle_{\gamma \in \{\text{Concat, Not, And}\}}$  whose definitions mirror those of the operations in our translation base  $\{H_\gamma\}_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$
- As desired,  $h^o k$  is a homomorphism from  $\langle E, K_\gamma \rangle_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$  (Disambiguated-Mini-English) to  $\langle B, G_\gamma \rangle_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$  (our derived semantic algebra).
- **Thus, thanks to our translation base T, we now have an interpretation for Mini-English.**

The Interpretation for Mini-English:  $\langle B, G_\gamma, h^o j \rangle_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$

- (i)  $\langle B, G_\gamma \rangle_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$  is the derived algebra guaranteed by (3)/(4),
- (ii)  $h$  is the meaning assignment determined by  $\langle B, G_\gamma \rangle_{\gamma \in \{\text{Concat, Not, And}\}}$
- (iii)  $j$  is the lexical translation function in our translation base  $T$ .

- Note that this structure will satisfy our general definition of an interpretation:
  - Because  $h^o k$  is a homomorphism, we know that for all  $\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}$ ,  $K_\gamma$  and  $G_\gamma$  are of the same arity.
  - $h^o j$  is a function from the basic expressions of Mini-English into B.

*Clearly, this result in (5)/(6) generalizes to all languages, allowing us to state the following general theorem...*

## (7) General Theorem on Indirect Interpretation

Let  $L$  and  $L'$  be languages such that there is an interpretation  $B$  for  $L'$  and a translation base  $T$  from  $L$  to  $L'$ . There is an interpretation  $B'$  for  $L$ .

*You can no doubt already see how (7) follows from what we've seen so far...*

*For those who are interested, we can give a more explicit proof of it as in (8).*

### (8) Proof of the General Theorem on Indirect Interpretation

- Let  $h$  be the meaning assignment determined by  $\mathbf{B}$ . Let  $k$  be the translation function determined by  $\mathbf{T}$ . Let  $j$  be the lexical translation function in  $\mathbf{T}$ .
- By definition,  $h$  is a homomorphism from the syntactic algebra of  $\mathbf{L}' < \mathbf{A}, F_\gamma >_{\gamma \in \Gamma}$  to the semantic algebra of  $\mathbf{B} < \mathbf{B}, G_\gamma >_{\gamma \in \Gamma}$
- By definition,  $k$  is a homomorphism from the syntactic algebra of  $\mathbf{L} < \mathbf{E}, K_\gamma >_{\gamma \in \Pi}$  to the algebra  $< \mathbf{A}, H_\gamma >_{\gamma \in \Pi}$  where  $\{H_\gamma\}_{\gamma \in \Pi}$  are all polynomial operations over  $< \mathbf{A}, F_\gamma >_{\gamma \in \Gamma}$
- Therefore, by the theorem in (3)/(4), there is an algebra  $< \mathbf{B}, G_\gamma >_{\gamma \in \Pi}$  such that  $h$  is also a homomorphism from  $< \mathbf{A}, H_\gamma >_{\gamma \in \Pi}$  to  $< \mathbf{B}, G_\gamma >_{\gamma \in \Pi}$
- Consequently,  $h \circ k$  is a homomorphism from  $< \mathbf{E}, K_\gamma >_{\gamma \in \Pi}$  to  $< \mathbf{B}, G_\gamma >_{\gamma \in \Pi}$
- Therefore, we know that for all  $\gamma \in \Pi$ ,  $K_\gamma$  and  $G_\gamma$  have the same arity.
- Moreover,  $h \circ j$  is a function from the basic categories in  $\mathbf{L}$  to  $\mathbf{B}$ .
- Therefore, the structure  $< \mathbf{B}, G_\gamma, h \circ j >_{\gamma \in \Pi}$  is an interpretation for  $\mathbf{L}$ .

### (9) Remark

Furthermore, the meaning assignment determined by the interpretation  $< \mathbf{B}, G_\gamma, h \circ j >_{\gamma \in \Pi}$  will be  $h \circ k$  (proof left as exercise for the student)

### (10) The Big Upshot

#### a. Our Initial Question:

Given our background theory of language and meaning, under what conditions can we guarantee that translating from one language  $L$  into another language  $L'$  gives us a compositional semantics for  $L$ .

#### b. Answer:

If we can provide an interpretation for  $L'$  and the translation from  $L$  to  $L'$  satisfy the conditions of a **translation base**, then we are guaranteed a compositional semantics for  $L$ .

Thus, a new viable path to providing a semantics for a (natural) language is to provide a **translation base** from that language to a logical language whose semantics is already defined.

## 2. Illustration: Mini-English and Politics-NoQ

### (11) First Observation

When we compose together  $k$  and  $h$  in (1), we get a mapping from sentences of Disambiguated-Mini-English to meanings in B.

Illustration: Let  $T$  be the tree such that  $R(T) = \text{Barack loves Michelle}$ .

- a.  $h^0k(T) =$  (by definition of function composition)
- b.  $h(k(T)) =$  (by definition of Mini-English)
- c.  $h(k(K_{\text{Merge-S}}(\langle \text{Barack}, \emptyset \rangle, K_{\text{Merge-IV}}(\langle \text{loves} \emptyset \rangle, \langle \text{Michelle}, \emptyset \rangle)))) =$  (by homomorphism prop.)
- d.  $h(H_{\text{Merge-S}}(k(\langle \text{Barack}, \emptyset \rangle), H_{\text{Merge-IV}}(k(\langle \text{loves} \emptyset \rangle), k(\langle \text{Michelle}, \emptyset \rangle)))) =$  (by definition of  $k$ )
- e.  $h(H_{\text{Merge-S}}(\text{barack}', H_{\text{Merge-IV}}(\text{loves}', \text{micelle}')))) =$  (by def. of  $H_{\text{Merge-IV}}$ )
- f.  $h(H_{\text{Merge-S}}(\text{barack}', (\text{loves}' \text{ micelle}')))) =$  (by definition of  $H_{\text{Merge-S}}$ )
- g.  $h((\text{loves}' \text{ micelle}') \text{ barack}') =$  (by definition of Politics-NoQ)
- h.  $h(F_{\text{Concat}}(F_{\text{Concat}}(\text{loves}', \text{micelle}'), \text{barack}')) =$  (by homomorphism prop.)
- i.  $G_{\text{Concat}}(G_{\text{Concat}}(h(\text{loves}'), h(\text{micelle}')), h(\text{barack}')) =$  (by def. of  $h$ )
- j.  $G_{\text{Concat}}(G_{\text{Concat}}(j, \text{Michelle}), \text{Barack}) =$  (by def. of  $G_{\text{Concat}}$ )
- k.  $G_{\text{Concat}}(j(\text{Michelle}), \text{Barack}) =$  (by def. of  $G_{\text{Concat}}$ )
- l.  $j(\text{Michelle})(\text{Barack}) =$  (by def. of  $j$ )
- m. 1

### (12) Second Observation

Observing the behavior of  $h^0k$  over a range of examples, we can directly construct an interpretation for Disambiguated-Mini-English, which will mirror the behavior of  $h^0k$

(13) **The Interpretation of Disambiguated Mini-English**

Let  $\langle B, G_\gamma, l \rangle_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$  be the structure defined as follows:

a. The Definition of the Set B

The set B is the same set as the set B in  $\mathbf{B} = \langle B, G_\gamma, f \rangle_{\gamma \in \{\text{Concat, Not, And}\}}$ , the interpretation of Politics-NoQ we had defined previously.

- That is  $B = \bigcup_{\tau \in T} D_\tau, \{\text{Michelle, Barack, Mitt}\}$

b. The Definition of the Semantic Operations

The operations  $\{G_\gamma\}_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$  are defined as follows:

$$(i) \quad G_{\text{Not}} = G_{\text{Not}} \quad (\text{defined previously})$$

$$(ii) \quad G_{\text{And}} = G_{\text{And}} \quad (\text{defined previously})$$

$$(iii) \quad G_{\text{If}} = G_{\text{Not}} \langle G_{\text{And}} \langle \text{Id}_{1,2}, G_{\text{Not}} \langle \text{Id}_{2,2} \rangle \rangle \rangle$$

$$(iv) \quad G_{\text{Merge-IV}} = G_{\text{Concat}} \quad (\text{defined previously})$$

$$(v) \quad G_{\text{Merge-S}} = G_{\text{Concat}} \langle \text{Id}_{2,2}, \text{Id}_{1,2} \rangle$$

Note:

- The definitions of the operations  $\{G_\gamma\}_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$  mirror the definitions of the operations  $\{H_\gamma\}_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$  in our translation base.
- Moreover,  $\{G_\gamma\}_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$  are all polynomial operations over the semantic algebra  $\langle B, G \rangle_{\gamma \in \{\text{Concat, Not, And}\}}$
- Consequently,  $\langle B, G \rangle_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$  is an algebra.

c. The Definition of the Lexical Interpretation Function

The lexical interpretation function  $l$  is defined as follows:

$$(i) \quad l(\langle \text{Barack}, \emptyset \rangle) = \text{Barack}$$

$$(ii) \quad l(\langle \text{Michelle}, \emptyset \rangle) = \text{Michelle}$$

$$(iii) \quad l(\langle \text{Mitt}, \emptyset \rangle) = \text{Mitt}$$

$$(iv) \quad l(\langle \text{smokes}, \emptyset \rangle) = \text{the function } h \text{ equal to } f(\text{smokes}')$$

$$(v) \quad l(\langle \text{loves}, \emptyset \rangle) = \text{the function } j \text{ equal to } f(\text{loves}')$$

Note: Where  $g$  is the meaning assignment determined by  $\mathbf{B}$ ,  $l = g \circ j$

(14) **Remark**  $\langle B, G_\gamma, l \rangle_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}}$  is an interpretation of Mini-English

(15) **Remark**

The meaning assignment  $g$  determined by  $\langle B, G_\gamma, I \rangle_\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}$  is equal to  $h^o k$

(16) **Illustration**

Let  $g$  be the meaning assignment determined by  $\langle B, G_\gamma, I \rangle_\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If}\}$ . Let  $T$  be the tree such that  $R(T) = \text{Barack loves Michelle}$ .

- a.  $g(T)$  = (by definition of Mini-English)
- c.  $g(K_{\text{Merge-S}}(\langle \text{Barack}, \emptyset \rangle, K_{\text{Merge-IV}}(\langle \text{loves} \emptyset \rangle, \langle \text{Michelle}, \emptyset \rangle))) =$   
(by homomorphism prop.)
- d.  $G_{\text{Merge-S}}(g(\langle \text{Barack}, \emptyset \rangle), G_{\text{Merge-IV}}(g(\langle \text{loves} \emptyset \rangle), g(\langle \text{Michelle}, \emptyset \rangle))) =$   
(by definition of  $g$ )
- e.  $G_{\text{Merge-S}}(\text{Barack}, G_{\text{Merge-IV}}(j, \text{Michelle})) =$  (by def. of  $G_{\text{Merge-IV}}$ )
- f.  $G_{\text{Merge-S}}(\text{Barack}, j(\text{Michelle})) =$  (by def. of  $G_{\text{Merge-S}}$ )
- g.  $j(\text{Michelle})(\text{Barack}) =$  (by def. of  $j$ )
- m. 1

### 3. Indirect Interpretation: A Summary

(17) **Direct Interpretation**

Let  $L$  be a language. **Direct interpretation of  $L$**  is the specification of an interpretation  $B$  of  $L$ .

(18) **Indirect Interpretation**

Let  $L$  be a language. **Indirect interpretation of  $L$**  is the specification of a language  $L'$ , an interpretation  $B'$  of  $L'$  and a translation base  $T$  from  $L$  to  $L'$ .

(19) **Indirect Interpretation Always Yields Direct Interpretation**

- Suppose that we have indirectly interpreted the language  $L$ .
  - That is, we have defined a language  $L'$  an interpretation  $B'$  of  $L'$  and a translation base  $T$  from  $L$  to  $L'$
- Given the result in (7)/(8) it is thus trivial to construct an interpretation  $B$  of  $L$ .

## (20) Some Choice Quotes

- From Halvorsen & Ladusaw (1979), p. 210:  
“An understanding of [the eliminability of indirect interpretation] is necessary to understand the use of...logics of PTQ and most other analyses within Montague grammar. As has been stated elsewhere, the [logical language] is an ***expository device and is in no way a necessary part of an analysis of any language offered within this theory.*** By using the easily interpreted [logical language] as a mediator, natural languages can be analyzed syntactically and then provided with a translation...from them to [the logical language] to induce their interpretation....***This method of analysis amounts to direct interpretation of natural language.***” (emphasis mine)
- From Dowty et al. (1981), p. 263:  
“Translating English into [a logical language] was therefore not essential to interpreting the English phrases we generated; ***it was simply a convenient intermediate step in assigning them meanings. This step could have been eliminated had we chosen to describe the interpretation of English directly...*** This point is important, because anyone who does not appreciate it may misunderstand the role of [logical languages] in applications of Montague’s descriptive framework to natural languages.” (emphasis mine)

## (21) Why Do Indirect Interpretation?

### It's Just Sometimes Conceptually 'Easier'

If the logical language is well-designed and familiar to readers, then it can provide a more ‘perspicuous’ representation (statement / name) of the meanings that we wish to be assigned to the (natural) language expressions.

#### *Illustration (From 610):*

‘[  $\lambda x_e : x \text{ smokes}$  ]’ vs. ‘The function  $f$  from  $D_e$  to  $D_t$  such that for all  $x \in D_e$ ,  $f(x) = 1$  iff  $x$  smokes.’

- From Halvorsen & Ladusaw (1979), p. 216:  
“Since the translation process is less involved than interpretation, presentation of fragments of the languages becomes clearer.”
- From Dowty et al. (1981), p. 264:  
“[The purpose of indirect interpretation] was to have a convenient, compact notation for giving a briefer statement of semantic rules than we were able to give in earlier chapters of this book, where semantic rules were formulated rather long-windedly in English... [The logical language] could provide us with names for meanings...”

(21) **Road Map of Where We've Been and Where We're Going**

- We've now covered the conceptual core of the Montague Grammar architecture, as well as the key sections of Montague's "Universal Grammar".
  - Sections of UG Covered: 1, 2, 3, 5
- What we *haven't* covered from UG is:
  - The general theory of Fregean interpretations
  - Montague's presentation of the syntax/semantics of his Intensional Logic
  - Montague's presentation of the syntax/translation of a fragment of English
- **However, those sections of UG are largely superceded by Montague's paper "The Proper Treatment of Quantification in Ordinary English" (PTQ)**
  - If we had all the time in the world, we would cover *both* systems and compare them (hint for final paper).
  - Given our limited time, however, I'd like to now move us from UG to PTQ.
- What's Next on the Agenda:  
Extending this system to handle quantification in FOL and English.
  - An algebraic treatment of FOL (with quantification)
  - An translation base from a fragment of English to FOL.
- *Following that, we'll examine Montague's Intensional Logic and its applications to English (and various puzzles therein).*

### Problem Set on Translation and Indirect Interpretation

#### (1) Polynomial Operations and Algebras

Let  $\mathbf{A}$  be an algebra  $\langle A, F_\gamma \rangle_{\gamma \in \Gamma}$ , and let  $H$  be a member of the polynomial operations  $K$  over  $\mathbf{A}$ . Show that  $A$  is closed under  $H$ .

*There are four steps to showing that  $A$  is closed under  $H$ , corresponding to the four ‘ways’ by which  $H$  could be a member of  $K$ .*

- a. Step One: Show that  $A$  is closed under  $F_\gamma$  for all  $\gamma \in \Gamma$
- b. Step Two:  
Let  $Id_{n,m}$  be any identity function (projection function). Show that  $A$  is closed under  $Id_{n,m}$ .
- c. Step Three: Let  $a \in A$ . Show that  $A$  is closed under  $C_{a,m}$ .
- d. Step Four:  
Let  $G$  be an  $n$ -ary function that  $A$  is closed under. Let  $F_1, \dots, F_n$  be  $n$   $m$ -ary functions that  $A$  is closed under. Show that  $A$  is closed under  $G \langle F_1, \dots, F_n \rangle$ .

#### (2) Derived Syntactic Rules and Meaningful Expressions

Let  $\mathbf{L}$  be a language  $\langle\langle A, F_\gamma, X_\delta, S, \delta_0 \rangle_\gamma \in \Gamma, \delta \in \Delta, R \rangle$ , and let  $\langle H, \langle \delta_1, \dots, \delta_n \rangle, \delta \rangle$  be a derived syntactic rule of  $\mathbf{L}$ . Show that if  $\varphi_1, \dots, \varphi_n$  are such that each  $\varphi_i \in C_{\delta_i}$ , then  $H(\varphi_1, \dots, \varphi_n) \in C_\delta$ .

*There are four steps to showing that  $H(\varphi_1, \dots, \varphi_n) \in C_\delta$ , corresponding to the four ‘ways’ by which  $\langle H, \langle \delta_1, \dots, \delta_n \rangle, \delta \rangle$  could be a derived syntactic rule of  $\mathbf{L}$ .*

- a. Step One:  
Let  $\langle H, \langle \delta_1, \dots, \delta_n \rangle, \delta \rangle \in S$ . Show that if  $\varphi_1, \dots, \varphi_n$  are such that each  $\varphi_i \in C_{\delta_i}$ , then  $H(\varphi_1, \dots, \varphi_n) \in C_\delta$ .
- b. Step Two:  
Let  $\langle H, \langle \delta_1, \dots, \delta_n \rangle, \delta \rangle$  be a rule of the form  $\langle Id_{n,m}, \langle \delta_1, \dots, \delta_n, \dots, \delta_m \rangle, \delta \rangle$ . Show that if  $\varphi_1, \dots, \varphi_m$  are such that each  $\varphi_i \in C_{\delta_i}$ , then  $Id_{n,m}(\varphi_1, \dots, \varphi_m) \in C_\delta$ .
- c. Step Three:  
Let  $\langle H, \langle \delta_1, \dots, \delta_n \rangle, \delta \rangle$  be of the form  $\langle C_{a,n}, \langle \delta_1, \dots, \delta_n \rangle, \delta \rangle$ , where  $a \in C_\delta$ . Show that if  $\varphi_1, \dots, \varphi_n$  are such that each  $\varphi_i \in C_{\delta_i}$ , then  $C_{a,n}(\varphi_1, \dots, \varphi_n) \in C_\delta$ .

*Continued on next page...*

d. Step Four:

Let the rule  $\langle F, \langle \delta_1, \dots, \delta_n \rangle, \delta \rangle$  have the property that if  $\varphi_1, \dots, \varphi_n$  are such that each  $\varphi_i \in C_{\delta_i}$ , then  $F(\varphi_1, \dots, \varphi_n) \in C_\delta$ . In addition, for each  $G_1, \dots, G_n$ , let the rule  $\langle G_j, \langle \delta'_1, \dots, \delta'_m \rangle, \delta_j \rangle$  have the property that if  $\varphi_1, \dots, \varphi_m$  are such that each  $\varphi_i \in C_{\delta'_i}$ , then  $G_j(\varphi_1, \dots, \varphi_m) \in C_{\delta_j}$ .

Show that the rule  $\langle F \langle G_1, \dots, G_n \rangle, \langle \delta'_1, \dots, \delta'_m \rangle, \delta \rangle$  has the property that if  $\varphi_1, \dots, \varphi_m$  are such that each  $\varphi_i \in C_{\delta'_i}$ , then  $F \langle G_1, \dots, G_n \rangle (\varphi_1, \dots, \varphi_m) \in C_\delta$ .

(3) **An Exercise in Indirect and Direct Interpretation of a Fragment of English**

- a. Minimally alter our language Mini-English so that its expressions now include strings like *Neither Mitt smokes nor Barack smokes*.

**Note:**

**Don't worry if your system also produces such marginal strings as *Neither it is not the case that Mitt smokes nor Barack loves Michelle*.**

- b. Take our translation base in (46)-(50) on the handout “The Notion of a Translation Base”, and minimally alter it so that strings like *Neither Mitt smokes nor Barack smokes* receive appropriate translations in Politics-NoQ.

**Note:**

**Be sure to show that any *new polynomial operations* in your translation base have the property in (40c) on handout “The Notion of a Translation Base.”**

- c. Please show how your new translation base, along with our interpretation for Politics-NoQ, assigns a truth-value to the analysis tree for *Neither Mitt smokes nor Barack smokes*.
- d. Given your proposed translation base, construct a direct interpretation of Disambiguated Mini-English, and show how it interprets *Neither Mitt smokes nor Michelle smokes*.

**Unit 5:**  
**An Algebraic Approach to Quantification and**  
**Lambda Abstraction**

## An Algebraic Approach to Quantification and Lambda Abstraction: Preliminaries<sup>1</sup>

### (1) Our Analytic Toolbox (The Framework)

#### a. A Theory of Syntax

- (i) ‘Disambiguated’ languages as quintuples  $\langle A, F_\gamma, X_\delta, S, \delta_0 \rangle_{\gamma \in \Gamma, \delta \in \Delta}$
- (ii) Languages as pairs  $\langle \langle A, F_\gamma, X_\delta, S, \delta_0 \rangle_{\gamma \in \Gamma, \delta \in \Delta}, R \rangle$

#### b. A Theory of Semantics

- (i) Interpretations as structures  $\langle B, G_\gamma, f \rangle_{\gamma \in \Gamma}$
- (ii) Meaning assignments as homomorphisms from  $\langle A, F_\gamma \rangle_{\gamma \in \Gamma}$  to  $\langle B, G_\gamma \rangle_{\gamma \in \Gamma}$

#### c. A Theory of Translations Inducing Interpretations

- (i) Translation bases as structures  $\langle g, H_\gamma, j \rangle_{\gamma \in \Gamma'}$
- (ii) Translations as homomorphisms from  $\langle A', F_\gamma \rangle_{\gamma \in \Gamma'}$  to  $\langle A, H_\gamma \rangle_{\gamma \in \Gamma'}$
- (iii) Translation from  $L'$  to  $L$  induces the interpretation  $\langle B, G'_\gamma, f' \rangle_{\gamma \in \Gamma'}$

We’re now going to apply these analytic tools (framework) to the phenomenon of quantification, both in English and in logical languages...

### (2) An Outline of the Plan

#### a. Step One: Introduce a logical language with quantification.

- Since we’re ultimately going to use this logical language to indirectly interpret English quantificational NPs, this language will also include ‘ $\lambda$ ’

#### b. Step Two: Introduce a model-theoretic semantics for the language

- This will give us a more comprehensible means for establishing certain key semantic properties of the language.
- It will also provide the basis for Step Three

#### c. Step Three: Introduce a (Montagovian) interpretation structure for the language

- **This is the intellectually hardest step.**

#### d. Step Four: Introduce a translation base from English to the logical language.

- **This takes work, but the key ideas will already be familiar to you.**

---

<sup>1</sup> These notes are based upon material in the following readings: Partee *et al.* (1993) Chapter 13, Dowty *et al.* (1981) Chapter 4, and Thomason (1974) Chapter 7 (Montague’s “Universal Grammar”).

## 1. The Logical Language TL: Non-Montagovian Presentation

I assume that everyone has a basic familiarity with lambda abstraction and its applications to natural language semantics.

- For a review, the student is referred to Heim & Kratzer (1998: 34-40) and Partee *et al.* (1993: 336-369).

### (3) Typed Logic (TL)

A typed logic (language) is a language whose vocabulary of symbols satisfies the conditions in (4), and whose syntax satisfies the conditions in (7).

### (4) The Vocabulary of a TL

#### a. The Logical Constants:<sup>2</sup>

- (i) Sentence Connectives:  $\sim, \&, \vee, \rightarrow$
- (ii) Quantifiers:  $\forall, \exists$
- (iii) Lambda Operator:  $\lambda$

#### b. The Syntactic Symbols: ( , )

#### c. The Non-Logical Constants:

##### (i) Constants:

For every type  $\tau \in T$ , a **countable** set of constants of type  $\tau$ ,  $CON_{\tau}$

##### (ii) Variables:

For every type  $\tau \in T$ , a **countably infinite** set of variables of type  $\tau$ :

$$VAR_{\tau} = \{ v_{\tau, n} : n \in \mathbb{N} \}$$

### (5) Remarks

- a. Note that in (4c), we are using the types to categorize expressions of our language, just as in our Montagovian definitions of FOL-NoQ and Politics-NoQ.
- b. Given (4ci), we will have individual constants ( $CON_e$ ) and for any  $n \in \mathbb{N}$ , a set of  $n$ -ary predicate constants ( $CON_{\langle e, t \rangle}$ ,  $CON_{\langle e, \langle e, t \rangle \rangle}$ ,  $CON_{\langle e \langle e \langle e, t \rangle \rangle \rangle}$ , ...)
- c. Given (4ci), we can also have constants of type  $t$ ,  $\langle \langle e, t \rangle, t \rangle$ ,  $\langle e, e \rangle$ , etc.
  - Given our intended applications, we won't make use of such constants here.
- d. Given (4cii), our variables now come with a subscript indicating their type.
  - $v_{e,3}$        $v_{\langle e, t \rangle, 4}$        $v_{\langle e, \langle e, t \rangle \rangle, 2}$        $v_{\langle \langle e, t \rangle, t \rangle, 6}$        $v_{t, 190}$

<sup>2</sup> We'll now make use of a larger set of primitive logical operators. This will make the statement of our semantics longer, but it will later simplify our analysis of English.

(6) **Meta-Language Abbreviations for Variables**

Although our variables ‘officially’ all look like those in (5d), to save space we will make use of the following meta-language abbreviations for variables of type  $e$  and  $\langle e, t \rangle$ :

a.  $x_n = v_{e,n}$

b.  $P_n = v_{\langle e, t \rangle, n}$

In our (relatively) informal syntaxes for PL, FOL, FOL-NoQ, Politics, and Politics-NoQ, our syntax defined a set of ‘WFFs’...

- But, now that our logical language contains ‘ $\lambda$ ’, it will be more efficient to arrange our syntax so that it defines sets of ‘meaningful expressions’ of various types  $\tau \in T(ME_\tau)$

(7) **The Syntax of a TL**

a. If  $\varphi \in CON_\tau$  or  $\varphi \in VAR_\tau$ , then  $\varphi \in ME_\tau$

b. If  $\varphi \in ME_{\langle \sigma, \tau \rangle}$  and  $\psi \in ME_\sigma$ , then  $(\varphi \psi) \in ME_\tau$

c. If  $\varphi, \psi \in ME_t$ , then

(i)	$\sim\varphi \in ME_t$
(ii)	$(\varphi \& \psi) \in ME_t$
(iii)	$(\varphi \vee \psi) \in ME_t$
(iv)	$(\varphi \rightarrow \psi) \in ME_t$

d. If  $v \in VAR_\tau$ , and  $\varphi \in ME_t$ , then

(i)	$\exists v \varphi \in ME_t$
(ii)	$\forall v \varphi \in ME_t$

e. If  $v \in VAR_\sigma$ , and  $\varphi \in ME_\tau$ , then  $(\lambda v \varphi) \in ME_{\langle \sigma, \tau \rangle}$

(8) **Illustration of a TL Language: Politics+ $\lambda$**

a. Vocabulary of Politics+ $\lambda$

(i) *Logical Constants:* (as in (4a))

(ii) *Syntactic Symbols:* (, )

(iii) *Non-Logical Constants*

1. *Variables* (as in (4cii))

2. *Constants:*  $CON_e = \{ \text{mitt}', \text{barack}', \text{michelle}' \}$

$CON_{\langle et \rangle} = \{ \text{smokes}', \text{man}', \text{president}' \}$

$CON_{\langle e, t \rangle} = \{ \text{loves}' \}$

**For all other  $\tau \in T$ ,  $CON_\tau = \emptyset$**

b. Syntax of Politics+ $\lambda$ : (as in (7))

(9) Some Illustrative Meaningful Expressions of Politics+ $\lambda$

a.  $\sim ((\text{loves}' \text{ mitt}') \text{ barack}')$

- (i)  $\text{loves}' \in \text{CON}_{\langle e, t \rangle}, \text{mitt}', \text{barack}' \in \text{CON}_e$  (8a)
- (ii)  $\text{loves}' \in \text{ME}_{\langle e, t \rangle}, \text{mitt}', \text{barack}' \in \text{ME}_e$  (7a)
- (iii)  $(\text{loves}' \text{ mitt}') \in \text{ME}_{\langle e, t \rangle}$  (7b)
- (iv)  $((\text{loves}' \text{ mitt}') \text{ barack}') \in \text{ME}_t$  (7b)
- (v)  $\sim ((\text{loves}' \text{ mitt}') \text{ barack}') \in \text{ME}_t$  (7c)

b.  $\forall x_3 ((\text{smokes}' x_3) \rightarrow \sim ((\text{loves}' \text{ mitt}') x_3))$

- (i)  $\text{loves}' \in \text{CON}_{\langle e, t \rangle}, \text{smokes}' \in \text{CON}_{\langle e, t \rangle}, \text{mitt}', \text{barack}' \in \text{CON}_e, x_3 \in \text{VAR}_e$  (8a)
- (ii)  $\text{loves}' \in \text{ME}_{\langle e, t \rangle}, \text{smokes}' \in \text{ME}_{\langle e, t \rangle}, \text{mitt}', \text{barack}', x_3 \in \text{ME}_e$  (7a)
- (iii)  $(\text{loves}' \text{ mitt}') \in \text{ME}_{\langle e, t \rangle}$  (7b)
- (iv)  $((\text{loves}' \text{ mitt}') x_3) \in \text{ME}_t$  (7b)
- (v)  $\sim ((\text{loves}' \text{ mitt}') x_3) \in \text{ME}_t$  (7c)
- (vi)  $(\text{smokes}' x_3) \in \text{ME}_t$  (7b)
- (vii)  $((\text{smokes}' x_3) \rightarrow \sim ((\text{loves}' \text{ mitt}') x_3)) \in \text{ME}_t$  (7c)
- (viii)  $\forall x_3 ((\text{smokes}' x_3) \rightarrow \sim ((\text{loves}' \text{ mitt}') x_3)) \in \text{ME}_t$  (7d)

c.  $\exists P_4 (P_4 \text{ michelle}')$

- (i)  $\text{michelle}' \in \text{CON}_e, P_4 \in \text{VAR}_{\langle e, t \rangle}$  (8a)
- (ii)  $\text{michelle}' \in \text{ME}_e, P_4 \in \text{ME}_{\langle e, t \rangle}$  (7a)
- (iii)  $(P_4 \text{ michelle}') \in \text{ME}_t$  (7b)
- (iv)  $\exists P_4 (P_4 \text{ michelle}') \in \text{ME}_t$  (7d)

d.  $((\lambda x_3 (\text{man}' x_3)) \text{ mitt}')$

- (i)  $\text{man}' \in \text{CON}_{\langle e, t \rangle}, \text{mitt}' \in \text{CON}_e, x_3 \in \text{VAR}_e$  (8a)
- (ii)  $\text{man}' \in \text{ME}_{\langle e, t \rangle}, \text{mitt}', x_3 \in \text{ME}_e$  (7a)
- (iii)  $(\text{man}' x_3) \in \text{ME}_t$  (7b)
- (iv)  $((\lambda x_3 (\text{man}' x_3)) \in \text{ME}_{\langle e, t \rangle}$  (7e)
- (v)  $((\lambda x_3 (\text{man}' x_3)) \text{ mitt}') \in \text{ME}_t$  (7b)

e.  $(\lambda P_4 \forall x_3 ((\text{man}' x_3) \rightarrow (P_4 x_3)))$

- (i)  $\text{man}' \in \text{CON}_{\langle e, t \rangle}, P_4 \in \text{VAR}_{\langle e, t \rangle}, x_3 \in \text{VAR}_e$  (8a)
- (ii)  $\text{man}', P_4 \in \text{ME}_{\langle e, t \rangle}, x_3 \in \text{ME}_e$  (7a)
- (iii)  $(\text{man}' x_3), (P_4 x_3) \in \text{ME}_t$  (7b)
- (iv)  $((\text{man}' x_3) \rightarrow (P_4 x_3)) \in \text{ME}_t$  (7c)
- (v)  $\forall x_3 ((\text{man}' x_3) \rightarrow (P_4 x_3)) \in \text{ME}_t$  (7d)
- (vi)  $(\lambda P_4 \forall x_3 ((\text{man}' x_3) \rightarrow (P_4 x_3))) \in \text{ME}_{\langle e, t \rangle, t \rangle}$  (7e)

## 2. The Semantics of a TL: Non-Montagovian Presentation

In this section, we will develop a model-theoretic semantics for TL languages.

- This model-theoretic semantics will provide the basis from which we will develop a (Montagovian) interpretation for a TL language.
- Moreover, the model-theoretic semantics will allow for a more perspicuous demonstration of a key validity in TL languages ('lambda conversion')
- Furthermore, seeing the relationship between this model-theoretic semantics and the later interpretation will lay important groundwork for our presentation of PTQ

### (10) The Denotations Based on a Set E<sup>3</sup>

Let T be the set of types and E be some non-empty set (of entities). If  $\tau \in T$ , then the set  $D_{\tau, E}$  of *denotations of type  $\tau$  based on E* is defined as follows:

- (i)  $D_{e, E} = E$
- (ii)  $D_{t, E} = \{ 0, 1 \}$
- (iii) If  $\sigma, \tau \in T$ , then  $D_{\langle \sigma, \tau \rangle, E} =$  the set of functions from  $D_{\sigma, E}$  to  $D_{\tau, E}$

### (11) Definition of a Model for a TL Language

A model  $\mathcal{M}$  for a TL language  $L$  is a pair  $\langle E, I \rangle$  consisting of:<sup>4</sup>

- a. A *non-empty* set E, called the 'domain of  $\mathcal{M}$ '
- b. A function I, whose domain is equal to (i) and whose range satisfies the condition in (ii).
  - (i) *Domain of I:*  $\bigcup_{\tau \in T} CON_{\tau}$
  - (ii) *Condition on Range of I:* If  $\alpha \in CON_{\tau}$ , then  $I(\alpha) \in D_{\tau, E}$

### (12) Remarks

If  $\langle E, I \rangle$  is a model for a TL language, then:

- a. If  $\alpha$  is an individual constant ( $\alpha \in CON_e$ ), then  $I(\alpha)$  is a member of E ( $D_{e, E}$ )
- b. If  $\alpha$  is an n-ary predicate letter ( $\alpha \in CON_{\langle e, \dots, t \rangle}$ ), then  $I(\alpha)$  is the curried characteristic function of an n-ary relation in E ( $D_{\langle e, \dots, t \rangle, E}$ )

<sup>3</sup> Note that since our model-theoretic semantics won't ever interpret 'syntactic garbage', we needn't add the special element **garbage** to  $D_{e, E}$ .

<sup>4</sup> To avoid confusion with the notation in (10), the domains of models will now generally be represented as 'E'.

(13) **Illustration: A Model for Politics+ $\lambda$**

Let the model  $\mathcal{M}$  be the pair  $\langle \{\text{Barack, Michelle, Mitt}\}, I \rangle$ , where  $I$  consists of the following mappings:

- a.  $I(\text{michelle}') = \text{Michelle}$
- b.  $I(\text{barack}') = \text{Barack}$
- c.  $I(\text{mitt}') = \text{Mitt}$
- d.  $I(\text{smokes}') = h = \{ \langle \text{Michelle}, 0 \rangle, \langle \text{Barack}, 1 \rangle, \langle \text{Mitt}, 0 \rangle \}$
- e.  $I(\text{man}') = i = \{ \langle \text{Michelle}, 0 \rangle, \langle \text{Barack}, 1 \rangle, \langle \text{Mitt}, 1 \rangle \}$
- f.  $I(\text{president}') = k = \{ \langle \text{Michelle}, 0 \rangle, \langle \text{Barack}, 1 \rangle, \langle \text{Mitt}, 0 \rangle \}$
- e.  $f(\text{loves}') = j = \begin{array}{c} \text{Michelle} \\ \text{Barack} \\ \text{Mitt} \end{array} \rightarrow \begin{cases} \text{Michelle} \rightarrow 1 \\ \text{Barack} \rightarrow 1 \\ \text{Mitt} \rightarrow 0 \end{cases} \quad \begin{cases} \text{Michelle} \rightarrow 1 \\ \text{Barack} \rightarrow 1 \\ \text{Mitt} \rightarrow 0 \end{cases} \quad \begin{cases} \text{Michelle} \rightarrow 0 \\ \text{Barack} \rightarrow 0 \\ \text{Mitt} \rightarrow 1 \end{cases}$

*Given that we now have variables for types other than  $e$ , we also need a concomitant change in our definition of a variable assignment...*

(14) **Variable Assignment**

Let  $\mathcal{M}$  be a model  $\langle E, I \rangle$  of a TL language. Then  $g$  is a *variable assignment* (based on  $\mathcal{M}$ ) if its domain is equal to (i) and its range satisfies the property in (ii).

- (i) *Domain of  $g$ :*  $\bigcup_{\tau \in T} \text{VAR}_\tau$
- (iii) *Condition on Range of  $g$ :* If  $\alpha \in \text{VAR}_\tau$ , then  $g(\alpha) \in D_{\tau, E}$

(15) **Remarks**

If  $\mathcal{M} = \langle E, I \rangle$  is a model for a TL language, and  $g$  is a variable assignment based on  $\mathcal{M}$

- a. If  $\alpha$  is an individual variable ( $\alpha \in \text{VAR}_e$ ), then  $g(\alpha)$  is a member of  $E$  ( $D_{e, E}$ )
- b. If  $\alpha$  is an  $n$ -ary predicate letter variable ( $\alpha \in \text{VAR}_{e, \dots, t}$ ), then  $g(\alpha)$  is the curried characteristic function of an  $n$ -ary relation in  $E$  ( $D_{e, \dots, t, E}$ )

With the definitions in (11) and (14), we can now define the notion of ‘interpretation with respect to a model  $\mathcal{M}$  and a variable assignment  $g$ ,  $[[.]]^{M,g}$ ’.

(16) **Interpretation With Respect to a Model and a Variable Assignment**

Let  $\mathcal{M}$  be a model  $\langle E, I \rangle$  for a TL language  $L$  and  $g$  be a variable assignment based on  $\mathcal{M}$ . The interpretation (a.k.a. denotation) of a meaningful expression of  $L$  relative to  $\mathcal{M}$  and  $g$ ,  $[[.]]^{M,g}$  is defined as follows:

- a. If  $v \in \bigcup_{\tau \in T} \text{VAR}_{\tau}$ , then  $[[v]]^{M,g} = g(v)$
- b. If  $\alpha \in \bigcup_{\tau \in T} \text{CON}_{\tau}$ , then  $[[\alpha]]^{M,g} = I(\alpha)$
- c. If  $\varphi = (\psi \chi)$ , then  $[[\varphi]]^{M,g} = [[\psi]]^{M,g}([[\chi]]^{M,g})$
- d. If  $\varphi = \sim\psi$ , then  $[[\varphi]]^{M,g} = 1 \quad iff \quad [[\psi]]^{M,g} = 0$
- e. If  $\varphi = (\psi \& \chi)$ , then  $[[\varphi]]^{M,g} = 1 \quad iff \quad [[\psi]]^{M,g} = 1 \text{ and } [[\chi]]^{M,g} = 1$
- f. If  $\varphi = (\psi \vee \chi)$ , then  $[[\varphi]]^{M,g} = 1 \quad iff \quad [[\psi]]^{M,g} = 1 \text{ or } [[\chi]]^{M,g} = 1$
- g. If  $\varphi = (\psi \rightarrow \chi)$ , then  $[[\varphi]]^{M,g} = 1 \quad iff \quad [[\psi]]^{M,g} = 0 \text{ or } [[\chi]]^{M,g} = 1$
- h. If  $\varphi = \exists v\psi$  and  $v \in \text{VAR}_{\tau}$ , then  $[[\varphi]]^{M,g} = 1 \quad iff$   
there is an  $a \in D_{\tau,E}$  such that  $[[\psi]]^{M,g(v/a)} = 1$
- i. If  $\varphi = \forall v\psi$  and  $v \in \text{VAR}_{\tau}$ , then  $[[\varphi]]^{M,g} = 1 \quad iff \quad \text{for all } a \in D_{\tau,E}, [[\psi]]^{M,g(v/a)} = 1$
- j. If  $\varphi = (\lambda v\psi)$ ,  $v \in \text{VAR}_{\sigma}$  and  $\psi \in \text{ME}_{\tau}$ , then  $[[\varphi]]^{M,g} =$   
The function  $p$  whose domain is  $D_{\sigma,E}$ , whose range is  $D_{\tau,E}$  and for all  $a \in D_{\sigma,E}$ ,  
 $p(a) = [[\psi]]^{M,g(v/a)}$

(17) **Remarks**

- a. Given the sorting of variables into types, along with the definition in (14), our definitions in (16h,i) allow our language to quantify, not only over entities, but **also over objects of all other types**.
- b. Similarly, given the definition in (14), our definition in (16j) entails that the type of the variable in a lambda expression will determine the domain of the function denoted by the expression.

I will now use the definitions in (16) to show how our model in (13) can be used to assign semantic values to meaningful expressions of Politics+ $\lambda$ .

- For reasons of space, the calculations below are greatly abbreviated...
- Students should work out for themselves the full calculations based on (16)...

(18) **Interpreting Expressions of Politics+ $\lambda$**

Let  $\mathcal{M}$  be the model defined in (13). Let  $g$  be some arbitrary variable assignment based on  $\mathcal{M}$ .

a.  $\forall x_3 (\text{man}' x_3)$

- (i)  $[[\forall x_3 (\text{man}' x_3)]]^{M,g} = 1 \quad \text{iff}$
- (ii) For all  $a \in D_{e,E}$ ,  $i(a) = 1 \quad \text{iff}$
- (iii) For all  $a \in \{\text{Michelle, Barack, Mitt}\}$ ,  
 $\{<\text{Michelle}, 0>, <\text{Barack}, 1>, <\text{Mitt}, 1>\}(a) = 1$
- (iv) Thus,  $[[\forall x_3 (\text{man}' x_3)]]^{M,g} = 0$

b.  $\exists P_4 (P_4 \text{ michelle}')$

- (i)  $[[\exists P_4 (P_4 \text{ michelle}')]]^{M,g} = 1 \text{ iff}$
- (ii) There is an  $a \in D_{e,E}$  such that  $a(\text{Michelle}) = 1$
- (iii) Thus,  $[[\exists P_4 (P_4 \text{ michelle}')]]^{M,g} = 1$

Note: Even though neither  $h$ ,  $i$ , nor  $k$  map Michelle to 1, there are still other functions  $f \in D_{e,E}$  such that  $f(\text{Michelle}) = 1$

c.  $(\lambda x_3 ((\text{man}' x_3) \& (\text{smokes}' x_3)))$

- (i)  $[[ (\lambda x_3 ((\text{man}' x_3) \& (\text{smokes}' x_3))) ]]]^{M,g} =$
- (ii) The function  $p$  with domain  $D_{e,E}$ , range  $D_{t,E}$ , and for all  $a \in D_{e,E}$ ,  
 $p(a) = 1 \text{ iff } i(a) = 1 \text{ and } h(a) = 1 =$
- (iv) The function  $p$  with domain  $\{\text{Mitt, Barack, Michelle}\}$ , range  $\{0,1\}$ , and  
for all  $a \in \{\text{Mitt, Barack, Michelle}\}$ ,  
 $p(a) = 1 \text{ iff } \{<\text{Michelle}, 0>, <\text{Barack}, 1>, <\text{Mitt}, 1>\}(a) = 1 \text{ and } \{<\text{Michelle}, 0>, <\text{Barack}, 1>, <\text{Mitt}, 0>\}(a) = 1 =$
- (v)  $\{<\text{Michelle}, 0>, <\text{Barack}, 1>, <\text{Mitt}, 0>\} =$
- (vi) **The characteristic function of the set of ‘men who smoke’**

- d.  $(\lambda P_4 (P_4 \text{ mitt}'))$
- (i)  $[[((\lambda P_4 (P_4 \text{ mitt}')))]^{M,g}] =$
  - (ii) The function  $p$  with domain  $D_{\langle et\rangle,E}$ , range  $D_{t,E}$  and for all  $a \in D_{\langle et\rangle,E}$ ,  
 $p(a) = a(\text{Mitt}) =$
  - (iii) **The function  $p$  with domain  $D_{\langle et\rangle,E}$ , range  $D_{t,E}$  and for all  $a \in D_{\langle et\rangle,E}$ ,**  
 $p(a) = 1$  iff  $a(\text{Mitt}) = 1$
  - (iv) **The characteristic function of the set of  $\langle et\rangle$ -functions  $f$  such that**  
 $f(\text{Mitt}) = 1$
  - (v) **The characteristic function of the set of ‘properties that Mitt has’.**

Note:

Given the result in (18d), we also have it that:

- $[[((\lambda P_4 (P_4 \text{ mitt}')) \text{ smokes}')]^{M,g}] = p([[\text{smokes}']])^{M,g} = p(h) = h(\text{Mitt}) = 0$
- $[[((\lambda P_4 (P_4 \text{ mitt}')) \text{ man}')]^{M,g}] = p([[\text{man}']])^{M,g} = p(i) = i(\text{Mitt}) = 1$

- e.  $(\lambda P_4 \forall x_3 ((\text{man}' x_3) \rightarrow (P_4 x_3)))$
- (i)  $[[((\lambda P_4 \forall x_3 ((\text{man}' x_3) \rightarrow (P_4 x_3))))]]^{M,g} =$
  - (ii) The function  $p$  with domain  $D_{\langle et\rangle,E}$ , range  $D_{t,E}$  and for all  $a \in D_{\langle et\rangle,E}$ ,  
 $p(a) = 1$  iff for all  $a' \in D_{e,E}$ , either  $i(a') = 0$  or  $a(a') = 1 =$
  - (iii) The function  $p$  with domain  $D_{\langle et\rangle,E}$ , range  $D_{t,E}$  and for all  $a \in D_{\langle et\rangle,E}$ ,  
 $p(a) = 1$  iff for all  $a' \in D_{e,E}$ , if  $i(a') = 1$  then  $a(a') = 1$
  - (iv) **The characteristic function of the set of  $\langle et\rangle$ -functions  $f$  such that if  $x$  is ‘a man’, then  $f(x) = 1$**
  - (v) **The characteristic function of the set of ‘properties every man has’**

Note:

Given the result in (18e), we also have it that:

- $[[((\lambda P_4 \forall x_3 ((\text{man}' x_3) \rightarrow (P_4 x_3))) \text{ smokes}')]^{M,g} = p([[\text{smokes}']])^{M,g} = p(h) = 0$
- $[[((\lambda P_4 \forall x_3 ((\text{man}' x_3) \rightarrow (P_4 x_3))) \text{ man}')]^{M,g} = p([[\text{man}']])^{M,g} = p(i) = 1$

### 3. A Key Logical Equivalence

The model-theoretic semantics introduced above yields a key logical equivalence, one that will be of **much** use to us shortly...

#### (19) Notational Preliminary

If  $\varphi, \psi$  are meaningful expressions of a TL language  $L$ ,  $\varphi \in ME_\tau$  and  $v \in VAR_\tau$ , then ' $[\varphi/v]\psi$ ' is the meaningful expression just like  $\psi$  except that every free instance of  $v$  is replaced with  $\varphi$ .

$$[\text{barack}'/x_3] ((\text{man}' x_3) \& (\text{smokes}' x_3)) = ((\text{man}' \text{barack}') \& (\text{smokes}' \text{barack}'))$$

$$[\text{smokes}'/P_4] \forall x_3 ((\text{man}' x_3) \rightarrow (P_4 x_3)) = \forall x_3 ((\text{man}' x_3) \rightarrow (\text{smokes}' x_3))$$

#### (20) Key Validity: Lambda Conversion

Let  $(\lambda v\psi)$  and  $\varphi$  be meaningful expressions with no variables in common, and let  $\varphi \in ME_\tau$  and  $v \in VAR_\tau$ . It follows that the meaningful expressions in (a) and (b) are logically equivalent:

$$\text{a. } ((\lambda v\psi) \varphi) \qquad \qquad \text{b. } [\varphi/v]\psi$$

#### (21) Illustration of Lambda Conversion

Each of the following pairs of meaningful expressions are logically equivalent.

$$\text{a. } ((\lambda x_3 ((\text{man}' x_3) \& (\text{smokes}' x_3))) \text{barack}') \Leftrightarrow ((\text{man}' \text{barack}') \& (\text{smokes}' \text{barack}'))$$

$$\text{b. } ((\lambda P_4 \forall x_3 ((\text{man}' x_3) \rightarrow (P_4 x_3))) \text{smokes}') \Leftrightarrow \forall x_3 ((\text{man}' x_3) \rightarrow (\text{smokes}' x_3))$$

#### (22) Informal Proof of Lambda Conversion

Let  $\mathcal{M}$  be a model  $\langle E, I \rangle$  of a TL language and  $g$  be a variable assignment based on  $\mathcal{M}$ .

- If  $(\lambda v\psi)$  is such that  $v \in VAR_\tau$  and  $\psi \in ME_\sigma$ , then by (16j),  $[[[(\lambda v\psi)]]^{M,g}$  is the function  $p$  with domain  $D_{\tau,E}$  and range  $D_{\sigma,E}$  such that for all  $a$  in  $D_{\tau,E}$ ,  $p(a) = [[\psi]]^{M,g(v/a)}$
- If  $\varphi \in ME_\tau$ , then by (16) it follows that there is an  $a \in D_{\tau,E}$  such that  $[[\varphi]]^{M,g} = a$ .
- By (16c), it follows that  $[[((\lambda v\psi) \varphi)]]^{M,g} = [[(\lambda v\psi)]]^{M,g}([[[\varphi]]^{M,g}]] = p(a) = [[\psi]]^{M,g(v/a)}$
- Finally, **it is intuitively the case that**  $[[\psi]]^{M,g(v/a)} = [[[\varphi/v]\psi]]^{M,g}$   
**Thus,**  $[[((\lambda v\psi) \varphi)]]^{M,g} = [[[\varphi/v]\psi]]^{M,g}$

(23) a. **Question:**  
Why must we assume in (20) that  $(\lambda v\psi)$  and  $\varphi$  have no variables in common?

b. **Answer:**  
If  $\varphi$  contains a free variable  $v'$  which is bound in  $\psi$ , then it can sometimes happen that  $v'$  is free in  $((\lambda v\psi)\varphi)$  but bound in  $[\varphi/v]\psi$ .

*Illustration:*  $((\lambda v_{t,3} \exists x_2((Px_2) \& v_{t,3})) (Qx_2))$       is not logically equivalent to:  
 $\exists x_2((Px_2) \& (Qx_2))$

## (24) The Utility of Lambda Conversion

As the semanticists in the audience are no doubt aware, we will be making much use of lambda conversion when it comes time to translate structures of English to structures of Politics+ $\lambda$ .

- Our translation process will output rather complex formulae of Politics+ $\lambda$ .
- Lambda conversion will allow us to ‘convert’ those complex formulae into simpler, logically equivalent formulae.
  - These simpler formulae will more transparently represent the ‘meanings’ that our translation ends up assigning to the expressions of English.

### Rough Illustration:

In the rough illustration below, imagine that  $h$  is our translation function from English to Politics+ $\lambda$ .

- (i)  $h(\text{every man smokes}) =$
- (ii)  $H_{\text{Merge-S}}( h(\text{every man}), h(\text{smokes}) ) =$
- (iii)  $H_{\text{Merge-S}}( (\lambda P_4 \forall x_3 ( (\text{man}' x_3) \rightarrow (P_4 x_3))), \text{smokes}' ) =$
- (iv)  $( (\lambda P_4 \forall x_3 ( (\text{man}' x_3) \rightarrow (P_4 x_3))) \text{smokes}' ) \Leftrightarrow$
- (v)  $\forall x_3((\text{man}' x_3) \rightarrow (\text{smokes}' x_3))$

#### 4. The Logical Language Politics+ $\lambda$ : Montagovian Presentation

In (8), we provide a relatively informal, non-Montagovian definition of the language Politics+ $\lambda$ .

- We will now represent Politics+ $\lambda$  as a ‘disambiguated language’, in the sense of Montague (1974).
- In the next handout, we will develop a Montagovian interpretation for Politics+ $\lambda$ , based upon this definition...

#### (25) The Vocabulary of Politics+ $\lambda$

##### a. The Logical Constants:

- |       |                              |                               |
|-------|------------------------------|-------------------------------|
| (i)   | <i>Sentence Connectives:</i> | $\sim, \&, \vee, \rightarrow$ |
| (ii)  | <i>Quantifiers:</i>          | $\forall, \exists$            |
| (iii) | <i>Lambda Operator:</i>      | $\lambda$                     |

##### b. The Syntactic Symbols: ( , )

##### c. The Non-Logical Constants:

###### (i) *Constants:*

$$\begin{aligned} \text{CON}_e &= \{ \text{'mitt'}, \text{'barack'}, \text{'michelle'} \} \\ \text{CON}_{\langle e \rangle} &= \{ \text{'smokes'}, \text{'man'}, \text{'president'} \} \\ \text{CON}_{\langle e \langle e, t \rangle \rangle} &= \{ \text{'loves'} \} \\ \text{For all other } \tau \in T, \text{CON}_\tau &= \emptyset \end{aligned}$$

###### (ii) *Variables:*

For every type  $\tau \in T$ , a **countably infinite** set of variables of type  $\tau$ :

$$\text{VAR}_\tau = \{ v_{\tau, n} : n \in \mathbb{N} \}$$

#### (26) The Syntactic Operations of Politics+ $\lambda$

- |    |                                    |   |                              |
|----|------------------------------------|---|------------------------------|
| a. | $F_{\text{Concat}}(\alpha, \beta)$ | = | $(\alpha \beta)$             |
| b. | $F_{\text{Not}}(\alpha)$           | = | $\sim\alpha$                 |
| c. | $F_{\text{And}}(\alpha, \beta)$    | = | $(\alpha \& \beta)$          |
| d. | $F_{\text{Or}}(\alpha, \beta)$     | = | $(\alpha \vee \beta)$        |
| e. | $F_{\text{If}}(\alpha, \beta)$     | = | $(\alpha \rightarrow \beta)$ |
| f. | $F_{\exists}(\alpha, \beta)$       | = | $\exists\alpha \beta$        |
| g. | $F_{\forall}(\alpha, \beta)$       | = | $\forall\alpha \beta$        |
| h. | $F_{\lambda}(\alpha, \beta)$       | = | $(\lambda\alpha \beta)$      |

(27) **The Syntactic Algebra of Politics+ $\lambda$**

$\langle A, F_\gamma \rangle_{\gamma \in \{\text{Concat, Not, And, Or, If, } \exists, \forall, \lambda\}}$  is the algebra such that:

- (i)  $\{ F_\gamma \}_{\gamma \in \{\text{Concat, Not, And, Or, If, } \exists, \forall, \lambda\}}$  are as defined in (26)
- (ii)  $A$  is the smallest set such that the following holds:

1. For all  $\tau \in T$ ,  $\text{CON}_\tau \subseteq A$  and  $\text{VAR}_\tau \subseteq A$
2.  $A$  is closed under  $\{ F_\gamma \}_{\gamma \in \{\text{Concat, Not, And, Or, If, } \exists, \forall, \lambda\}}$

Note:

Thus, the set  $A$  contains all the constants and variables in (25), and is closed under the syntactic operations in (26).

(28) **The Syntactic Category Labels of Politics+ $\lambda$**

Let  $T$  be the set of types.  $\Delta = T \cup \{ \langle \text{var}, \tau \rangle : \tau \in T \}$

Note:

Thus, the syntactic category labels include (i) all the types, and (ii) for every type  $\tau \in T$ , the ‘variable type’  $\langle \text{var}, \tau \rangle$ .

(29) **The Basic Expressions of Politics+ $\lambda$**

For every type  $\tau \in T$ :

- a.  $X_{\langle \text{var}, \tau \rangle} = \text{VAR}_\tau$
- b.  $X_\tau = \text{CON}_\tau \cup X_{\langle \text{var}, \tau \rangle}$

Note: Thus, it follows that:

- $X_{\langle \text{var}, e \rangle} = \{ v_{e,n} : n \in \mathbb{N} \} = \{ x_n : n \in \mathbb{N} \}$
- $X_e = \{ \text{mitt}', \text{barack}', \text{michelle}' \} \cup \{ x_n : n \in \mathbb{N} \}$
- $X_{\langle \text{var}, \langle e, t \rangle \rangle} = \{ v_{\langle e, t \rangle, n} : n \in \mathbb{N} \} = \{ P_n : n \in \mathbb{N} \}$
- $X_{\langle e, t \rangle} = \{ \text{smokes}', \text{man}', \text{president}' \} \cup \{ P_n : n \in \mathbb{N} \}$
- $X_{\langle \text{var}, \langle e, \langle e, t \rangle \rangle \rangle} = \{ v_{\langle e, \langle e, t \rangle \rangle, n} : n \in \mathbb{N} \}$
- $X_{\langle e, \langle e, t \rangle \rangle} = \{ \text{loves}' \} \cup \{ v_{\langle e, \langle e, t \rangle \rangle, n} : n \in \mathbb{N} \}$
- For all other types  $\tau \in T$ ,  $X_{\langle \text{var}, \tau \rangle} = \text{VAR}_\tau = \{ v_{\tau, n} : n \in \mathbb{N} \} = X_\tau$

(30) **The Syntactic Rules of Politics+ $\lambda$**

The (countably infinite) set S consists of (a) and (b) below:

- a. The following triples:
  - (i)  $\langle F_{\text{Not}}, \langle t \rangle, t \rangle$
  - (ii)  $\langle F_{\text{And}}, \langle t, t \rangle, t \rangle$
  - (iii)  $\langle F_{\text{Or}}, \langle t, t \rangle, t \rangle$
  - (vi)  $\langle F_{\text{If}}, \langle t, t \rangle, t \rangle$
- b. For every  $\sigma, \tau \in T$ , a triple of the following form:
  - (i)  $\langle F_{\text{Concat}}, \langle \langle \sigma, \tau \rangle, \sigma \rangle, \tau \rangle$
  - (ii)  $\langle F_{\exists}, \langle \langle \text{var}, \sigma \rangle, t \rangle, t \rangle$
  - (iii)  $\langle F_{\forall}, \langle \langle \text{var}, \sigma \rangle, t \rangle, t \rangle$
  - (iv)  $\langle F_{\lambda}, \langle \langle \text{var}, \sigma \rangle, \tau \rangle, \langle \sigma, \tau \rangle \rangle$

Note: Rules (30bii), (30bihi), and (30biv) can be informally read as the following, each of which mirrors an informal syntactic rule in (7d,e).

- “Applying  $F_{\exists}$  to a variable of category  $\langle \text{var}, \sigma \rangle$  and an expression of category  $t$  forms an expression of category  $t$ .”
- “Applying  $F_{\forall}$  to a variable of category  $\langle \text{var}, \sigma \rangle$  and an expression of category  $t$  forms an expression of category  $t$ .”
- “Applying  $F_{\lambda}$  to a variable of category  $\langle \text{var}, \sigma \rangle$  and an expression of category  $\tau$  forms an expression of category  $\langle \sigma, \tau \rangle$ .”

(31) **The Disambiguated Language Politics+ $\lambda$**

Politics+ $\lambda$  is the structure  $\langle A, F_\gamma, X_\delta, S, t \rangle_{\gamma \in \{\text{Concat, Not, And, Or, If, } \exists, \forall, \lambda\}, \delta \in \Delta}$  such that  $A, \{F_\gamma\}_{\gamma \in \{\text{Concat, Not, And, Or, If, } \exists, \forall, \lambda\}}, \{X_\delta\}_{\delta \in \Delta}, S, \Delta$  are all as defined in (25)-(30)

Note: Politics+ $\lambda$  is a disambiguated language.

We will now illustrate the definitions above by examining some meaningful expressions of (31), with associated calculations.

(32) **Illustrative ME:**  $\sim((\text{loves' mitt'})x_3) \in C_t$

- (i)  $\text{loves'} \in X_{e,t}, \text{mitt'}, x_3 \in X_e$  (by (29))
- (ii)  $\text{loves'} \in C_{e,t}, \text{mitt'}, x_3 \in C_e$  (by def. of  $\{C_\delta\}_{\delta \in \Delta}$ )
- (iii)  $\langle F_{\text{Concat}}, \langle e, \langle e, t \rangle, e \rangle, \langle e, t \rangle \rangle \in S$  (by (30))
- (iv)  $F_{\text{Concat}}(\text{loves'}, \text{mitt'}) \in C_{e,t}$  (by def. of  $\{C_\delta\}_{\delta \in \Delta}$ )
- (v)  $(\text{loves'} \text{ mitt'}) \in C_{e,t}$  (by def. of  $F_{\text{Concat}}$ )
- (vi)  $\langle F_{\text{Concat}}, \langle e, t \rangle, e \rangle, t \rangle \in S$  (by (30))
- (vii)  $F_{\text{Concat}}((\text{loves'} \text{ mitt'}), x_3) \in C_t$  (by def. of  $\{C_\delta\}_{\delta \in \Delta}$ )
- (viii)  $((\text{loves'} \text{ mitt'})x_3) \in C_t$  (by def. of  $F_{\text{Concat}}$ )
- (ix)  $\langle F_{\text{Not}}, \langle t \rangle, t \rangle \in S$  (by (30))
- (x)  $F_{\text{Not}}(((\text{loves'} \text{ mitt'})x_3)) \in C_t$  (by def. of  $\{C_\delta\}_{\delta \in \Delta}$ )
- (xi)  $\sim((\text{loves'} \text{ mitt'})x_3) \in C_t$  (by def. of  $F_{\text{Not}}$ )

(33) **Illustrative ME:**  $\forall x_3 ((\text{smokes'} x_3) \rightarrow \sim((\text{loves'} \text{ mitt'})x_3)) \in C_t$

- (i)  $\text{smokes'} \in X_{e,t}, x_3 \in X_e, x_3 \in X_{\text{var},e}$  (by (29))
- (ii)  $\text{smokes'} \in C_{e,t}, x_3 \in C_e, x_3 \in C_{\text{var},e}$  (by def. of  $\{C_\delta\}_{\delta \in \Delta}$ )
- (iii)  $\langle F_{\text{Concat}}, \langle e, t \rangle, e \rangle, t \rangle \in S$  (by (30))
- (iv)  $F_{\text{Concat}}(\text{smokes'}, x_3) \in C_t$  (by def. of  $\{C_\delta\}_{\delta \in \Delta}$ )
- (v)  $(\text{smokes'} x_3) \in C_t$  (by def. of  $F_{\text{Concat}}$ )
- (vi)  $\sim((\text{loves'} \text{ mitt'})x_3) \in C_t$  (by (32))
- (vii)  $\langle F_{\text{If}}, \langle t, t \rangle, t \rangle \in S$  (by (30))
- (viii)  $F_{\text{If}}((\text{smokes'} x_3), \sim((\text{loves'} \text{ mitt'})x_3)) \in C_t$  (by def. of  $\{C_\delta\}_{\delta \in \Delta}$ )
- (ix)  $((\text{smokes'} x_3) \rightarrow \sim((\text{loves'} \text{ mitt'})x_3)) \in C_t$  (by def. of  $F_{\text{If}}$ )
- (x)  $\langle F_{\forall}, \langle \text{var}, e \rangle, t \rangle, t \rangle \in S$  (by (30))
- (xi)  $F_{\forall}(x_3, ((\text{smokes'} x_3) \rightarrow \sim((\text{loves'} \text{ mitt'})x_3))) \in C_t$  (by def. of  $\{C_\delta\}_{\delta \in \Delta}$ )
- (xii)  $\forall x_3 ((\text{smokes'} x_3) \rightarrow \sim((\text{loves'} \text{ mitt'})x_3)) \in C_t$  (by def. of  $F_{\forall}$ )

(34) **Illustrative ME:**  $\exists P_4 (P_4 \text{ michelle'}) \in C_t$

- (i)  $\text{michelle'} \in X_e, P_4 \in X_{e,t}, P_4 \in X_{\text{var}, e,t}$  (by (29))
- (ii)  $\text{michelle'} \in C_e, P_4 \in C_{e,t}, P_4 \in C_{\text{var}, e,t}$  (by def. of  $\{C_\delta\}_{\delta \in \Delta}$ )
- (iii)  $\langle F_{\text{Concat}}, \langle e, t \rangle, e \rangle, t \rangle \in S$  (by (30))
- (iv)  $F_{\text{Concat}}(P_4, \text{michelle'}) \in C_t$  (by def. of  $\{C_\delta\}_{\delta \in \Delta}$ )
- (v)  $(P_4 \text{ michelle'}) \in C_t$  (by def. of  $F_{\text{Concat}}$ )
- (vi)  $\langle F_{\exists}, \langle \text{var}, e \rangle, t \rangle, t \rangle \in S$  (by (30))
- (vii)  $F_{\exists}(P_4, (P_4 \text{ michelle'})) \in C_t$  (by def. of  $\{C_\delta\}_{\delta \in \Delta}$ )
- (viii)  $\exists P_4 (P_4 \text{ michelle'}) \in C_t$  (by def. of  $F_{\exists}$ )

(35) **Illustrative ME:**  $((\lambda x_3 (\text{man}' x_3)) \text{ mitt}') \in C_t$

- (i)  $\text{man}' \in X_{<e,t>} \text{, mitt}' \in X_e \text{, } x_3 \in X_{<\text{var},e>}$  (by (29))
- (ii)  $\text{man}' \in C_{<e,>} \text{, mitt}' \in C_e \text{, } x_3 \in C_{<\text{var},e>}$  (by def. of  $\{C_\delta\}_{\delta \in \Delta}$ )
- (iii)  $< F_{\text{Concat}}, <<e,t>, e>, t> \in S$  (by (30))
- (iv)  $F_{\text{Concat}}(\text{man}', x_3) \in C_t$  (by def. of  $\{C_\delta\}_{\delta \in \Delta}$ )
- (v)  $(\text{man}' x_3) \in C_t$  (by def. of  $F_{\text{Concat}}$ )
- (vi)  $< F_\lambda, <<\text{var},e>, t>, <e, t>> \in S$  (by (30))
- (vii)  $F_\lambda(x_3, (\text{man}' x_3)) \in C_{<e,t>}$  (by def. of  $\{C_\delta\}_{\delta \in \Delta}$ )
- (viii)  $(\lambda x_3 (\text{man}' x_3)) \in C_{<e,>}$  (by def. of  $F_\lambda$ )
- (ix)  $F_{\text{Concat}}((\lambda x_3 (\text{man}' x_3)), \text{ mitt}') \in C_t$  (by def. of  $\{C_\delta\}_{\delta \in \Delta}$ )
- (x)  $((\lambda x_3 (\text{man}' x_3)) \text{ mitt}') \in C_t$  (by def. of  $F_{\text{Concat}}$ )

(36) **Illustrative ME:**  $(\lambda P_4 \forall x_3 ((\text{man}' x_3) \rightarrow (P_4 x_3))) \in C_{<<e,t>,t>}$

- (i)  $\text{man}' \in X_{<e,t>} \text{, } x_3 \in X_e \text{, } X_{<\text{var},e>} \text{, } P_4 \in X_{<e,>} \text{, } X_{<\text{var},<e,t>>}$  (by (29))
- (ii)  $\text{man}' \in C_{<e,t>} \text{, } x_3 \in C_e \text{, } C_{<\text{var},e>} \text{, } P_4 \in C_{<e,>} \text{, } C_{<\text{var},<e,t>>}$  (by def. of  $\{C_\delta\}_{\delta \in \Delta}$ )
- (iii)  $< F_{\text{Concat}}, <<e,t>, e>, t> \in S$  (by (30))
- (iv)  $F_{\text{Concat}}(\text{man}', x_3) \in C_t$  (by def. of  $\{C_\delta\}_{\delta \in \Delta}$ )
- (v)  $(\text{man}' x_3) \in C_t$  (by def. of  $F_{\text{Concat}}$ )
- (vi)  $F_{\text{Concat}}(P_4, x_3) \in C_t$  (by def. of  $\{C_\delta\}_{\delta \in \Delta}$ )
- (vii)  $(P_4 x_3) \in C_t$  (by def. of  $F_{\text{Concat}}$ )
- (viii)  $< F_{\text{If}}, <t,t>, t> \in S$  (by (30))
- (ix)  $F_{\text{If}}((\text{man}' x_3), (P_4 x_3)) \in C_t$  (by def. of  $\{C_\delta\}_{\delta \in \Delta}$ )
- (x)  $((\text{man}' x_3) \rightarrow (P_4 x_3)) \in C_t$  (by def. of  $F_{\text{If}}$ )
- (xi)  $F_\forall(x_3, ((\text{man}' x_3) \rightarrow (P_4 x_3))) \in C_t$  (by (30))
- (xii)  $\forall x_3 ((\text{man}' x_3) \rightarrow (P_4 x_3)) \in C_t$  (by def. of  $F_\forall$ )
- (xiii)  $< F_\lambda, <<\text{var},<e,t>>, t>, <<e,t>, t>> \in S$  (by (30))
- (xiv)  $F_\lambda(P_4, \forall x_3 ((\text{man}' x_3) \rightarrow (P_4 x_3))) \in C_{<<e,t>,t>}$  (by def. of  $\{C_\delta\}_{\delta \in \Delta}$ )
- (xv)  $(\lambda P_4 \forall x_3 ((\text{man}' x_3) \rightarrow (P_4 x_3))) \in C_{<<e,t>,t>}$  (by def. of  $F_\lambda$ )

## An Algebraic Approach to Quantification and Lambda Abstraction: Fregean Interpretations<sup>1</sup>

### (1) The Disambiguated Language Politics+λ

Politics+λ is the disambiguated language  $\langle A, F_\gamma, X_\delta, S, t \rangle_{\gamma \in \{\text{Concat, Not, And, Or, If, } \exists, \forall, \lambda\}, \delta \in \Delta}$  such that

#### a. Non-Logical Vocabulary

##### (i) *Constants:*

$$\begin{aligned} \text{CON}_e &= \{ \text{'mitt'}, \text{'barack'}, \text{'michelle'} \} \\ \text{CON}_{\langle e,t \rangle} &= \{ \text{'smokes'}, \text{'man'}, \text{'president'} \} \\ \text{CON}_{\langle e \langle e,t \rangle \rangle} &= \{ \text{'loves'} \} \\ \text{For all other } \tau \in T, \text{CON}_\tau &= \emptyset \end{aligned}$$

##### (ii) *Variables:*

For every type  $\tau \in T$ , a **countably infinite** set of variables of type  $\tau$ :

$$\text{VAR}_\tau = \{ v_{\tau,n} : n \in \mathbb{N} \}$$

#### b. Syntactic Algebra:

$\langle A, F_\gamma \rangle_{\gamma \in \{\text{Concat, Not, And, Or, If, } \exists, \forall, \lambda\}}$  is the algebra such that:

(i)  $\{ F_\gamma \}_{\gamma \in \{\text{Concat, Not, And, Or, If, } \exists, \forall, \lambda\}}$  are as defined on the previous handout

(ii)  $A$  is the smallest set such that the following holds:

1. For all  $\tau \in T$ ,  $\text{CON}_\tau \subseteq A$  and  $\text{VAR}_\tau \subseteq A$

2.  $A$  is closed under  $\{ F_\gamma \}_{\gamma \in \{\text{Concat, Not, And, Or, If, } \exists, \forall, \lambda\}}$

#### c. Syntactic Categories: $\Delta = T \cup \{ \langle \text{var}, \tau \rangle : \tau \in T \}$

#### d. Basic Expressions: For every type $\tau \in T$ :

$$(i) \quad X_{\langle \text{var}, \tau \rangle} = \text{VAR}_\tau$$

$$(ii) \quad X_\tau = \text{CON}_\tau \cup X_{\langle \text{var}, \tau \rangle}$$

#### e. Syntactic Rules:

$S$  is the (countably) infinite set consisting of:

(i)  $\langle F_{\text{Not}}, \langle \cdot, \cdot \rangle, t \rangle$

(ii)  $\langle F_{\text{And}}, \langle \cdot, \cdot \rangle, t \rangle$

(iii)  $\langle F_{\text{Or}}, \langle \cdot, \cdot \rangle, t \rangle$

(vi)  $\langle F_{\text{If}}, \langle \cdot, \cdot \rangle, t \rangle$

And, for every  $\sigma, \tau \in T$ , a triple of the following form:

(i)  $\langle F_{\text{Concat}}, \langle \langle \sigma, \tau \rangle, \sigma \rangle, \tau \rangle$  (ii)  $\langle F_{\exists}, \langle \langle \text{var}, \sigma \rangle, t \rangle, t \rangle$

(iii)  $\langle F_{\forall}, \langle \langle \text{var}, \sigma \rangle, t \rangle, t \rangle$  (iv)  $\langle F_{\lambda}, \langle \langle \text{var}, \sigma \rangle, \tau \rangle, \langle \sigma, \tau \rangle \rangle$

---

<sup>1</sup> These notes are based upon material in the following readings: Thomason (1974) Chapter 7 (Montague's "Universal Grammar").

(2) **Our Present Goal:** Define an interpretation for Politics+ $\lambda$ .

(3) **The Crucial Challenge**

To have an interpretation  $\mathbf{B} = \langle B, G_\gamma, f \rangle_{\gamma \in \{\text{Concat, Not, And, Or, If, } \exists, \forall, \lambda\}}$  for Politics+ $\lambda$ , we need to state some semantic (algebraic) operations  $G_\exists$ ,  $G_\forall$ , and  $G_\lambda$ , which will ‘correspond’ with  $F_\exists$ ,  $F_\forall$ , and  $F_\lambda$ . **What in the world could those be?**

(4) **The Plan**

- As we did several weeks ago for FOL, we’re going to see our way to a solution by reflecting informally on the meaning of deictic elements (akin to free pronouns).
- This will bring us to Montague’s definition of a **Fregean Interpretation**
- We’ll use **part of** this definition to set up an interpretation structure for Politics+ $\lambda$ .

---

## 1. Some Reflections on the Nature of Meaning

(5) **Illustrative Sentence with Free (Deictic) Pronoun:** He smokes.

(6) a. **Stupid Question**

In a context where I’m pointing to Barack, does sentence (5) have a meaning?

b. **Obvious Answer:**

Sure; after all, in such a context, (5) seems to have the intension in (i) and the extension in (ii).

- |   |  |
|---|--|
| (i) <i>Intension of (5), Pointing to Barack:</i>  | [ $\lambda w : \text{Barack smokes in } w$ ] |
| (ii) <i>Extension of (5), Pointing to Barack:</i> | 1 (true)                                     |

(7) a. **Difficult Question**

Suppose I don’t specify a context for (5). *Do we still say it has a meaning?*

b. **Negative Answer**

Without specifying a context, (5) doesn’t have a defined intension or extension. Thus, sentence (5) seems to be meaningless.

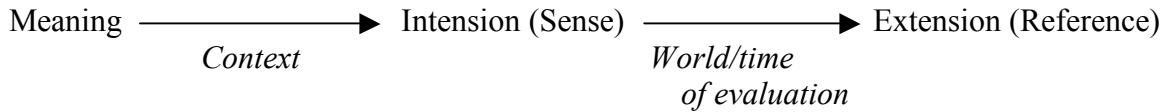
c. **Positive Answer**

To say that (5) has no meaning in such circumstances would fail to distinguish between it and gibberish like “snoochie boochies”.

- **Thus, its potential to have an intension/extension (relative to a context) should be viewed as constitutive of its meaning.**

## (8) A General Picture of Meaning, Fitting the Positive Answer

The ‘meaning’ – broadly construed – of an expression is a function from contexts to intensions. (*cf.* Kaplan’s notion of “character” vs. “content”).



- Some expressions have a *meaning* that maps every context to the same intension.
    - We can say that such expressions have a meaning that is ‘not context-dependent’

*Illustration:* Barack smokes.

Meaning: Function that maps every context  $c$  to  
[ $\lambda w : \text{Barack smokes in } w$ ]

- Some expressions have a *meaning* that maps different contexts to different intensions.
    - We can say that such expressions have a ‘context-dependent’ meaning.

*Illustration:* I smoke.

Meaning: Function that maps every context  $c$  to  
[ $\lambda w : \text{the speaker of } c \text{ smokes in } w$ ]

*Illustration:* He smokes.

Meaning: Function that maps every context  $c$  to  
 $[\lambda w : \text{the deictic focus of } c \text{ smokes in } w]$

## (9) Where We're Going From This

- We're going now to try to model meanings (in B) as functions from contexts (variable assignments) to intensions (functions from possible worlds to extensions).
  - We'll begin by redefining our system of types and denotations, so that they now include objects that can serve as 'intensions'.

## (10) The Types (New Definition)

The set  $T$  of types is the smallest set such that:

- a.  $e \in T$  (entities)
  - b.  $t \in T$  (truth-values)
  - c. If  $\sigma \in T$  and  $\tau \in T$ , then  $\langle\sigma, \tau\rangle \in T$  (functions from  $\sigma$  to  $\tau$ )
  - d. If  $\sigma \in T$ , then  $\langle s, \sigma \rangle \in T$  (**functions from world-times to  $\sigma$** )

(11) **The Denotations (New Definition)**

Let  $T$  be the set of types, let  $E$  be some non-empty set (of entities), **and let  $I$  be some other non-empty set (of ‘indices’ / world-time pairs).** The set  $D_{\tau, E, I}$  of *denotations of type  $\tau$  based on  $E$  and  $I$*  is defined as follows:

- a.  $D_{e,E,I} = E \cup \{\text{garbage}\}$
- b.  $D_{t,E,I} = \{0, 1\}$
- c. If  $\sigma, \tau \in T$  then  $D_{\langle \sigma, \tau \rangle, E, I} =$  the set of functions from  $D_{\sigma, E, I}$  to  $D_{\tau, E, I}$
- d. If  $\sigma \in T$  then  $D_{\langle s, \sigma \rangle, E, I} =$  **the set of functions from  $I$  to  $D_{\sigma, E, I}$**   
 $= (D_{\sigma, E, I})^I$

(12) **Remark**

The only real change to our earlier definition for the set of ‘denotations’ comes in (11d), which yields the ‘intensional objects’, including the following:

- a.  $D_{\langle s, e \rangle, E, I} =$  Functions from  $I$  (world-time pairs) to  $E$  (entities).  
 $=$  Individual Concepts
- b.  $D_{\langle s, t \rangle, E, I} =$  Functions from  $I$  (world-time pairs) to  $\{0, 1\}$   
 $=$  Propositions
- c.  $D_{\langle s, \langle e, t \rangle \rangle, E, I} =$  Functions from  $I$  (world-time pairs) to  $D_{\langle e, t \rangle, E, I}$   
 $=$  Properties

(13) **Some Considerations Towards ‘Meanings’**

- Following (8) and (9), we want ‘meanings’ to be functions from contexts (variable assignments) to intensions (functions from world-times to the denotations in (11)).
- Thus, if  $J$  is our set of contexts (variable assignments) and  $I$  is our set of world-time pairs, then a meaning should be something from the following set:

$$((\cup_{\tau \in T} D_{\tau, E, I})^I)^J$$

*Functions from  $J$  to functions from  $I$  to the set of all possible denotations*

- But recall from our first week that – due to the nature of currying – we can easily shift between the following two objects:  $(A^B)^C$  and  $A^{B \times C}$
- Thus, we can – and Montague does – state that meanings are elements from the following set:

$$((\cup_{\tau \in T} D_{\tau, E, I})^{I \times J})^{I \times J}$$

*Functions from the pairs in  $(I \times J)$  to the set of all possible denotations*

(14) **The Set of Possible Meanings**

Let  $T$  be the set of types, let  $E$ ,  $I$  and  $J$  be non-empty sets ( $E$  = entities;  $I$  = indices/world-times;  $J$  = contexts/variable-assignments). The set  $M_{\tau, E, I, J}$  of *meanings of type  $\tau$  based on  $E$ ,  $I$ , and  $J$*  is  $(D_{\tau, E, I})^{I \times J}$

Note: In UG, Montague sometimes shifts freely between  $(D_{\tau, E, I})^{I \times J}$  and  $((D_{\tau, E, I})^I)^J$

*With this conception of meaning at hand, we can begin to define Montague's notion of a 'Fregean Interpretation'...*

*A key ingredient in that definition is the notion of a 'type assignment'.....*

(15) **Type Assignment**

Let  $T$  be the set of types in (10). Let  $L$  be a language  $\langle\langle A, F_\gamma, X_\delta, S, \delta_0 \rangle\rangle_{\gamma \in \Gamma, \delta \in \Delta, R}$ . A type assignment for  $L$  is a function  $\sigma: \Delta \rightarrow T$  such that  $\sigma(\delta_0) = t$ .

Note: Thus, a type assignment pairs each syntactic category label  $\delta$  with exactly one type  $\tau$ , and it ensures that  $\delta_0$  (the category label for 'declarative sentences') is paired with type  $t$ .

(16) **The Core Idea Behind a Fregean Interpretation**

Informally speaking, Montague's Fregean interpretations are intended to be interpretations  $B = \langle B, G_\gamma, f \rangle_{\gamma \in \Gamma}$  such that:

- a.  $B$  is a set of 'meanings', as defined in (14); i.e.,  $B \subseteq \bigcup_{\tau \in T} M_{\tau, E, I, J}$
- b. **Expressions of the same category are assigned meanings of the same type.**
  - If  $g$  is the meaning assignment determined by  $B$ , then for each syntactic category  $C_\delta$ , there is a single type  $\tau$  such that if  $\varphi \in C_\delta$ , then  $g(\varphi) \in M_{\tau, E, I, J}$

(17) **Question:** Why have this condition in (16b)?

- Right from the start, (16b) was criticized as too restrictive. For example, it entails that "Barack" and "some man" must have the same type of denotation (and so the former cannot directly refer to Barack).
- To my knowledge, the restriction in (16b) doesn't 'do' anything.
  - It isn't necessary for the meaning assignment  $g$  to be a homomorphism...
  - It's just stipulated in UG and PTQ; there isn't much independent motivation

(18) **Possible Answer (?)**

It makes the mapping between the syntax and semantics akin to that of a typed logic?

(19) **Official Formal Definition of a Fregean Interpretation**

Let  $\mathbf{L}$  be a language  $\langle\langle A, F_\gamma, X_\delta, S, \delta_0 \rangle\rangle_{\gamma \in \Gamma, \delta \in \Delta, R}$ . A Fregean interpretation  $\mathbf{B}$  of  $\mathbf{L}$  is an interpretation  $\langle B, G_\gamma, f \rangle_{\gamma \in \Gamma}$  of  $\mathbf{L}$  such that for some type assignment  $\sigma$  and non-empty sets  $E, I, J$ :

$$a. \quad B \subseteq \bigcup_{\tau \in T} M_{\tau, E, I, J}$$

Note: Thus, the semantic values in  $B$  will all be ‘meanings’ in the sense of (14).

Thus, the meaning assignment based on  $\mathbf{B}$  maps expressions of  $\mathbf{L}$  to

- (i) functions from  $\langle \text{context}, \text{world-time} \rangle$  pairs to  $\bigcup_{\tau \in T} D_{\tau, E, I}$
- (ii) functions from contexts to

functions from world-time pairs to  $\bigcup_{\tau \in T} D_{\tau, E, I}$

- This clearly guarantees us property (16a).

$$b. \quad \text{For all } \delta \in \Delta, \text{ if } \varphi \in X_\delta, \text{ then } f(\varphi) \in M_{\sigma(\delta), E, I, J}$$

Note:

That is, if category  $\delta$  is paired with type  $\tau$  by the type assignment  $\sigma$ , then the lexical interpretation function  $f$  maps every basic expression of category  $\delta$  to a meaning of type  $\tau$ .

- This, combined with the next condition, guarantees property (16b).

$$c. \quad \text{If } \langle F_\gamma, \langle \delta_1, \dots, \delta_n \rangle, \delta \rangle \in S, \text{ and } b_1, \dots, b_n \text{ are such that for all } i, b_i \in M_{\sigma(\delta_i), E, I, J}, \text{ then } G_\gamma(b_1, \dots, b_n) \in M_{\sigma(\delta), E, I, J}.$$

Note:

That is, if categories  $\delta_1, \dots, \delta_n, \delta$  are paired with types  $\tau_1, \dots, \tau_n, \tau$  by  $\sigma$ , then

If the result of applying syntactic operation  $F_\gamma$  to expressions of category  $\delta_1, \dots, \delta_n$  is an expression of  $C_\delta$ , then

The result of applying the corresponding semantic operation  $G_\gamma$  to  
meanings of type  $\tau_1, \dots, \tau_n$  is a meaning of type  $\tau$ .

- This, combined with the preceding condition, guarantees (16b).

Note: We’ll also want in (19) to ensure that  $f$  and  $g$  do not end up mapping any meaningful expressions of  $\mathbf{L}$  to **garbage**, but we’ll leave the details of that implicit in what follows...

(20) **Concomitant Definition**

A Fregean interpretation for  $\mathbf{L}$  connected with  $E, I, J, \sigma$  is an interpretation  $\mathbf{B}$  such that the conditions in (19a-c) hold.

## 2. A ‘Partly-Fregean’ Interpretation for Politics+ $\lambda$

Now, in Politics+ $\lambda$  there are no ‘intensional’ environments...

*Thus, we won’t be making full use of the definition in (19) to interpret Politics+ $\lambda$ ...*

Let us, then, develop our own notion of a ‘Partly-Fregean’ interpretation, which will be quite applicable to our language in (1)...

### (21) The Key Ideas Behind a ‘Partly-Fregean’ Interpretation

Since we won’t be dealing with intensions for now, let us view meanings as functions from contexts (variable assignments) to denotations.

$$\text{Meaning} \xrightarrow{\quad\quad\quad} \text{Extension (Reference)}$$

*Context (Variable Assignment)*

#### a. The Types (For Politics+ $\lambda$ ):

The set T of types is the smallest set such that:

- (i)  $e \in T$  (entities)
- (ii)  $t \in T$  (truth-values)
- (iii) If  $\sigma \in T$  and  $\tau \in T$ , then  $\langle \sigma, \tau \rangle \in T$  (functions from  $\sigma$  to  $\tau$ )

#### b. The Possible Denotations (For Politics+ $\lambda$ ):

Let T be the set of types and E be some non-empty set (of entities). The set  $D_{\tau, E}$  of *denotations of type  $\tau$  based on E* is defined as follows:

- (i)  $D_{e, E} = E \cup \{\text{garbage}\}$
- (ii)  $D_{t, E} = \{0, 1\}$
- (iii) If  $\sigma, \tau \in T$ , then  $D_{\langle \sigma, \tau \rangle, E} =$  the set of functions from  $D_{\sigma, E}$  to  $D_{\tau, E}$

#### c. The Possible Meanings (For Politics+ $\lambda$ ):

Let T be the set of types, and let E and J be non-empty sets (E = entities; J = contexts/variable-assignments). The set  $M_{\tau, E, J}$  of *meanings of type  $\tau$  based on E and J* is  $(D_{\tau, E})^J$

### (22) Partly-Fregean Interpretation

Let L be a language  $\langle\langle A, F_\gamma, X_\delta, S, \delta_0 \rangle\rangle_{\gamma \in \Gamma, \delta \in \Delta, R}$ . A partly-Fregean interpretation B of L is an interpretation  $\langle B, G_\gamma, f \rangle_{\gamma \in \Gamma}$  of L such that for some type assignment  $\sigma$  and non-empty sets E, J:

- a.  $B \subseteq \bigcup_{\tau \in T} M_{\tau, E, J}$
- b. For all  $\delta \in \Delta$ , if  $\varphi \in X_\delta$ , then  $f(\varphi) \in M_{\sigma(\delta), E, J}$
- c. If  $\langle F_\gamma, \langle \delta_1, \dots, \delta_n \rangle, \delta \rangle \in S$ , and  $b_1, \dots, b_n$  are such that for all i,  $b_i \in M_{\sigma(\delta_i), E, J}$ , then  $G_\gamma(b_1, \dots, b_n) \in M_{\sigma(\delta), E, J}$ .

Note: Our definition of a ‘partly-Fregean’ interpretation simply takes Montague’s definition of a Fregean interpretation (19) and removes the set I (indices / world-time pairs).

(23) **Concomitant Definition**

A partly-Fregean interpretation for  $\mathbf{L}$  connected with  $E, J, \sigma$  is an interpretation  $\mathbf{B}$  such that the conditions in (22a-c) hold.

(24) **Goal:** Define a partly-Fregean interpretation for  $\text{Politics} + \lambda$ , where  $J$  is the set of possible variable assignments.

(25) **Immediate Issue:**

- Up to now, variable assignments have been defined with respect to a given model.
- Thus, we need a new definition of ‘variable assignment’, one that makes no reference to a model...

(26) **New Definition of Variable Assignment (for  $\text{Politics} + \lambda$ )**

Let  $E$  be a non-empty set (of entities). A *variable assignment based on  $E$*  is a function  $g$  satisfying the conditions below:

- (i) *Domain of  $g$ :*  $\bigcup_{\tau \in T} \text{VAR}_\tau$
- (ii) *Range of  $g$ :* If  $\alpha \in \text{VAR}_\tau$ , then  $g(\alpha) \in D_{\tau, E}$  (and  $g(\alpha) \neq \text{garbage}$ )

Note: Variable assignment map variables directly to *denotations*, not to *meanings*.

(27) **The Central, Scary Question that We’ve Been Putting Off Until Now**

How do we build a semantics for  $\text{Politics} + \lambda$ , where the semantic values (‘meanings’) are functions from variable assignments to denotations?

(28) **Some Informal Considerations to Get Us Started**

- a. The meaning of a constant maps every variable assignment to the same denotation
  - Just like the English name *Barack*, the individual constant **barack**’ should map every variable assignment to the same individual, Barack
    - After all, in our model-theoretic semantics,  $[[\text{barack}']]^{M,g}$  doesn’t depend on  $g$ ...
- b. The meaning of a variable can map different variable assignments to different denotations.
  - After all, in our model-theoretic semantics,  $[[x_3]]^{M,g}$  does depend on  $g$

(29) **Formalizing the Informal Considerations**

- a. If  $\varphi \in \text{CON}_\tau$ , then it should be that  $f(\varphi)(g) = f(\varphi)(g')$ , for all  $g, g' \in J$ .
- b. If  $\varphi \in \text{VAR}_\tau$ , then it should be that  $f(\varphi)(g) = g(\varphi)$  for all  $g \in J$ .

Note: If (29) holds, then

- (i) The meaning of a constant maps every variable assignment  $g$  to the same denotation.
- (ii) The meaning of a variable  $v$  maps a variable assignment to  $g(v)$ .  
The meaning of  $v$  can map different variable assignments to different denotations.

*Let's build on the ideas in (28)-(29), to define a special sub-class of partly-Fregean interpretations...*

(30) **Logically Possible Partly-Fregean Interpretation of Politics+ $\lambda$** <sup>2</sup>

A logically possible partly-Fregean interpretation of Politics+ $\lambda$  is a partly-Fregean interpretation  $\mathbf{B} = \langle B, G_\gamma, f \rangle_{\gamma \in \{\text{Concat, Not, And, Or, If, } \exists, \forall, \lambda\}}$  for Politics+ $\lambda$  connected with  $E$ ,  $J$ ,  $\sigma$ , such that:

- a.  $J$  is the set of variable assignments based on  $E$ .
- b. If  $\varphi \in \text{CON}_\tau$ , then  $f(\varphi)(g) = f(\varphi)(g')$ , for all  $g, g' \in J$ .
- c. If  $\varphi \in \text{VAR}_\tau$ , then  $f(\varphi)(g) = g(\varphi)$  for all  $g \in J$ .
- d. .... To be filled in later ...

*Finally, to complete the definition in (30), we will specify what the semantic operations in  $\{G_\gamma\}_{\gamma \in \{\text{Concat, Not, And, Or, If, } \exists, \forall, \lambda\}}$  must be...*

*This, of course, is the most challenging part...*

- Rather than motivate these operations from the ‘ground up’, I’m simply going to introduce them and show you how they work...
- Note that the set  $\mathbf{B} = \bigcup_{\tau \in T} M_{\tau, E, J}$  is closed under each of the following operations  $G_\gamma$ .

<sup>2</sup> The term in (30) and its definition mimic Montague’s definition in UG of a ‘logically possible Fregean interpretation’ of his Intensional Logic.

(31) The Operation  $G_{Concat}$

- a. If  $\alpha \in M_{\langle \sigma, \tau \rangle, E, J}$  and  $\beta \in M_{\sigma, E, J}$ , and  $\beta \neq \text{garbage}$ , then  $G_{Concat}(\alpha, \beta) =$   
The function  $F\text{-}A$  such that if  $g \in J$ , then  $F\text{-}A(g) = \alpha(g)(\beta(g))$
- b. Otherwise,  $G_{Concat}(\alpha, \beta) = \text{garbage}$

(32) How Does This Work?

- Consider the following two sentences: **(smokes' barack')**    **(smokes'  $x_3$ )**
- Since  $F_{Concat}$  and  $G_{Concat}$  should ‘correspond’, it follows for meaning assignment  $h$ :

$$\begin{array}{lcl} h((\text{smokes'} \text{ barack'})) & = & h((\text{smokes'} x_3)) \\ h(F_{Concat}(\text{smokes'}, \text{ barack'})) & = & h(F_{Concat}(\text{smokes'}, x_3)) \\ G_{Concat}(h(\text{smokes'}), h(\text{barack'})) & = & G_{Concat}(h(\text{smokes'}), h(x_3)) \\ G_{Concat}(f(\text{smokes'}), f(\text{barack'})) & = & G_{Concat}(f(\text{smokes'}), f(x_3)) \end{array} = = = =$$

- Now, given (22), in a partly-Fregean interpretation of Politics+ $\lambda$ :

$$f(\text{smokes'}) \in M_{\langle e, t \rangle, E, J} \quad f(\text{barack'}), f(x_3) \in M_{e, E, J}$$

- Thus, given (31a), it follows that:

$$G_{Concat}(f(\text{smokes'}), f(\text{barack'})) = \\ \text{The function } F\text{-}A \text{ such that if } g \in J, \text{ then } F\text{-}A(g) = f(\text{smokes'})(g)(f(\text{barack'})(g))$$

$$G_{Concat}(f(\text{smokes'}), f(x_3)) = \\ \text{The function } F\text{-}A' \text{ such that if } g \in J, \text{ then } -F\text{-}A'(g) = f(\text{smokes'})(g)(f(x_3)(g))$$

- Since **smokes'** and **barack'** are constants, and  $x_3$  is a variable, (30b,c) entails that:
  - For any  $g, g' \in J$ ,  $f(\text{smokes'})(g) = f(\text{smokes'})(g') = s \in D_{\langle e, t \rangle, E}$
  - For any  $g, g' \in J$ ,  $f(\text{barack'})(g) = f(\text{barack'})(g') = b \in D_{e, E}$
  - For any  $g \in J$ ,  $f(x_3)(g) = g(x_3)$
- Thus, it therefore follows that:

$$h((\text{smokes'} \text{ barack'})) = G_{Concat}(f(\text{smokes'}), f(\text{barack'})) = \\ \text{The function } F\text{-}A \text{ such that if } g \in J, \text{ then } F\text{-}A(g) = s(b)$$

$$h((\text{smokes'} x_3)) = G_{Concat}(f(\text{smokes'}), f(x_3)) = \\ \text{The function } F\text{-}A' \text{ such that if } g \in J, \text{ then } F\text{-}A'(g) = s(g(x_3))$$

(33) **Remarks**

- Thus, the ‘meaning’ of (**smokes’ barack’**) is a constant function from contexts (variable assignments) to truth-values:
  - It maps every variable assignment to the value that  $s$  yields for  $b$ .
- Thus, the ‘meaning’ of (**smokes’  $x_3$** ) is *not* a constant function from contexts (variable assignments) to truth-values:
  - It maps every variable assignment to the value that  $s$  yields for  $g(x_3)$
- **Parallel with Demonstratives:**
  - The ‘meaning’ of *Barack smokes* maps every context to the same intension
  - The ‘meaning’ of *He smokes* maps different contexts to different intensions
- **Parallel with Models:**
  - No matter what variable assignment  $g$  is chosen  $[(\text{smokes’ barack’})]^{M,g}$  will be the same value:  $I(\text{smokes’})(I(\text{barack’}))$
  - The value of  $[(\text{smokes’ } x_3)]^{M,g}$  will equal  $I(\text{smokes})(g(x_3))$ , and so will vary with the variable assignment  $g$ .

(34) **The Operation  $G_{Not}$**

- a. If  $\alpha \in M_{t, E, J}$  then  $G_{Not}(\alpha) =$   
The function  $Neg$  such that if  $g \in J$ , then  $Neg(g) = 1$  iff  $\alpha(g) = 0$
- b. Otherwise,  $G_{Not}(\alpha) = \text{garbage}$

(35) **How Does This Work? (Part 1)**

- Consider the following two sentences:  $\sim(\text{smokes’ barack’}) \sim(\text{smokes’ } x_3)$
- Since  $F_{Not}$  and  $G_{Not}$  should ‘correspond’, it follows for meaning assignment  $h$ :

$\begin{array}{rcl} h(\sim(\text{smokes’ barack’})) & = & \\ h(F_{Not}((\text{smokes’ barack’}))) & = & \\ G_{Not}(h(\text{smokes’ barack’})) & = & \end{array}$	$\begin{array}{rcl} h(\sim(\text{smokes’ } x_3)) & = & \\ h(F_{Not}((\text{smokes’ } x_3))) & = & \\ G_{Concat}(h(\text{smokes’ } x_3)) & = & \end{array}$
--	--

- Now, recall from (32) that  $h(\text{smokes’ barack’}), h((\text{smokes’ } x_3)) \in M_{t, E, J}$   
Thus, it follows from (34) that:

$$G_{Not}(h(\text{smokes’ barack’})) =$$

The function  $Neg$  such that if  $g \in J$ ,  $Neg(g) = 1$  iff  $h(\text{smokes’ barack’})(g) = 0$

$$G_{Concat}(h(\text{smokes’ } x_3)) =$$

The function  $Neg'$  such that if  $g \in J$ ,  $Neg'(g) = 1$  iff  $h(\text{smokes’ } x_3)(g) = 0$

### (36) How Does This Work? (Part 2)

- Now, also recall from (32) that:  

$$h((\text{smokes}' \text{ barack}')) = \text{The function } F\text{-}A \text{ such that if } g \in J, \text{ then } F\text{-}A(g) = s(b)$$

$$h((\text{smokes}' x_3)) = \text{The function } F\text{-}A' \text{ such that if } g \in J, \text{ then } F\text{-}A'(g) = s(g(x_3))$$
- It thus follows that:

$$h(\sim(\text{smokes}' \text{ barack})) = G_{\text{Not}}(h(\text{smokes}' \text{ barack})) = \\ \text{The function } Neg \text{ such that if } g \in J, Neg(g) = 1 \text{ iff } s(b) = 0$$

$$h(\sim(\text{smokes}' x_3)) = G_{\text{Concat}}(h(\text{smokes}' x_3)) = \\ \text{The function } Neg' \text{ such that if } g \in J, Neg'(g) = 1 \text{ iff } s(g(x_3)) = 0$$

### (37) Remarks

- Thus, the ‘meaning’ of  $\sim(\text{smokes}' \text{ barack})$  is a constant function from contexts (variable assignments) to truth-values:
  - It maps every variable assignment to 1 iff  $s(b) = 0$
- Thus, the ‘meaning’ of  $\sim(\text{smokes}' x_3)$  is *not* a constant function from contexts (variable assignments) to truth-values:
  - It maps every variable assignment to 1 iff  $s(g(x_3)) = 0$
- Parallel with Demonstratives:**
  - The ‘meaning’ of *Barack doesn’t smoke* maps every context to the same intension
  - The ‘meaning’ of *He doesn’t smoke* maps different contexts to different intensions
- Parallel with Models:**
  - No matter what variable assignment  $g$  is chosen  $[[\sim(\text{smokes}' \text{ barack})]]^{M,g}$  will be the same value: 1 iff  $I(\text{smokes}')(I(\text{barack}')) = 0$
  - The value of  $[[\sim(\text{smokes}' x_3)]]^{M,g}$  will equal 1 iff  $I(\text{smokes})(g(x_3)) = 0$ , and so will vary with the variable assignment  $g$ .

### (38) The Operation $G_{\text{And}}$

- If  $\alpha \in M_{t, E, J}$  and  $\beta \in M_{t, E, J}$ , then  $G_{\text{And}}(\alpha, \beta) =$   

$$\text{The function } Conj \text{ such that if } g \in J, \text{ then } Conj(g) = 1 \text{ iff } \alpha(g) = \beta(g) = 1$$
- Otherwise,  $G_{\text{And}}(\alpha) = \text{garbage}$

(39) How Does This Work?

- Consider the following two conjunctions:  
 $((\text{smokes' barack'}) \ \& \ (\text{smokes' mitt'}))$        $((\text{smokes' barack'}) \ \& \ (\text{smokes' } x_3))$
- Since  $F_{\text{And}}$  and  $G_{\text{And}}$  should ‘correspond’, it follows for meaning assignment  $h$ :

$$h(((\text{smokes' barack'}) \ \& \ (\text{smokes' mitt'}))) = \\ G_{\text{And}}( h((\text{smokes' barack'})), h((\text{smokes' mitt'})) )$$

$$h(((\text{smokes' barack'}) \ \& \ (\text{smokes' } x_3))) = \\ G_{\text{And}}( h((\text{smokes' barack'})), h((\text{smokes' } x_3)) )$$

- Now, recall that  $h(\text{smokes' barack'})$ ,  $h(\text{smokes' mitt'})$ ,  $h((\text{smokes' } x_3)) \in M_{t, e, J}$   
Thus, it follows from (38) that:

$$G_{\text{And}}( h((\text{smokes' barack'})), h((\text{smokes' mitt'})) ) = \\ \text{The function } Conj \text{ such that if } g \in J, \text{ then } Conj(g) = 1 \text{ iff} \\ h((\text{smokes' barack'}))(g) = h((\text{smokes' mitt'}))(g) = 1$$

$$G_{\text{And}}( h((\text{smokes' barack'})), h((\text{smokes' } x_3)) ) = \\ \text{The function } Conj \text{ such that if } g \in J, \text{ then } Conj(g) = 1 \text{ iff} \\ h((\text{smokes' barack'}))(g) = h((\text{smokes' } x_3))(g) = 1$$

- Next, recall that from (32) – and a parallel computation – that:  
 $h((\text{smokes' barack'})) =$  The function  $F\text{-}A$  such that if  $g \in J$ , then  $F\text{-}A(g) = s(b)$   
 $h((\text{smokes' mitt'})) =$  The function  $F\text{-}A$  such that if  $g \in J$ , then  $F\text{-}A(g) = s(m)$   
 $h((\text{smokes' } x_3)) =$  The function  $F\text{-}A'$  such that if  $g \in J$ , then  $F\text{-}A'(g) = s(g(x_3))$
- Consequently:

$$h(((\text{smokes' barack'}) \ \& \ (\text{smokes' mitt'}))) = \\ \text{The function } Conj \text{ such that if } g \in J, \text{ then } Conj(g) = 1 \text{ iff } s(b) = s(m) = 1$$

$$h(((\text{smokes' barack'}) \ \& \ (\text{smokes' } x_3))) = \\ \text{The function } Conj \text{ such that if } g \in J, \text{ then } Conj(g) = 1 \text{ iff } s(b) = s(g(x_3)) = 1$$

(40) Remarks

- Again, we have it that  $h(((\text{smokes' barack'}) \ \& \ (\text{smokes' mitt'})))$  is a constant function from variable assignment to truth-values.
  - Just as  $[[((\text{smokes' barack'}) \ \& \ (\text{smokes' mitt'}))]]^{M,g}$  doesn’t vary with  $g$
- Again, we have it that  $h(((\text{smokes' barack'}) \ \& \ (\text{smokes' } x_3)))$  is *not* a constant function from variable assignment to truth-values.
  - Just as  $[[((\text{smokes' barack'}) \ \& \ (\text{smokes' } x_3))]]^{M,g}$  does vary with  $g$

Given the discussion in (39)-(40), it can easily be seen that the following operations will similarly serve as semantic correlates of  $F_{Or}$  and  $F_{If}$

(41) **The Operation  $G_{Or}$**

- a. If  $\alpha \in M_{t, E, J}$  and  $\beta \in M_{t, E, J}$ , then  $G_{Or}(\alpha, \beta) =$   
The function  $Disj$  such that if  $g \in J$ , then  $Disj(g) = 1$  iff  $\alpha(g) = 1$  or  $\beta(g) = 1$
- b. Otherwise,  $G_{Or}(\alpha) = \mathbf{garbage}$

(42) **The Operation  $G_{If}$**

- a. If  $\alpha \in M_{t, E, J}$  and  $\beta \in M_{t, E, J}$ , then  $G_{If}(\alpha, \beta) =$   
The function  $Cond$  such that if  $g \in J$ , then  $Cond(g) = 1$  iff  $\alpha(g) = 0$  or  $\beta(g) = 1$
- b. Otherwise,  $G_{If}(\alpha) = \mathbf{garbage}$

Now, we finally come to the central problem of how to treat the variable-binding operators...

(43) **The Operation  $G_{\exists}$**

- a. If there is a type  $\tau \in T$  and a variable  $v \in VAR_{\tau}$  such that for all  $g \in J$ ,  
 $\alpha(g) = g(v)$ , and  $\beta \in M_{t, E, J}$ , then  
 $G_{\exists}(\alpha, \beta) =$  The function  $E$  such that if  $g \in J$ ,  $E(g) = 1$  iff  
there is an  $x \in D_{\tau, E}$  such that  $\beta(g(v/x)) = 1$
- b. Otherwise,  $G_{\exists}(\alpha, \beta) = \mathbf{garbage}$

(44) **How Does This Work? (Part 1)**

- First, note the requirement that there be a type  $\tau \in T$  and a variable  $v \in VAR_{\tau}$  such that for all  $g \in J$ ,  $\alpha(g) = g(v)$ .
- Given our conditions in (30b,c), this will only ever hold (in a logically possible partly-Fregean interpretation) if  $\alpha = f(v)$ , i.e., the ‘meaning’ of the variable  $v$ .
- Thus, this condition ensures that if the first argument of  $G_{\exists}$  is not a variable-meaning, then the output will be **garbage**.
  - Thus, our system will map syntactic garbage like ‘ $\exists$ mitt’ (*smokes*’ mitt’) to **garbage**

(45) **How Does This Work? (Part 2)**

- Now, consider the simple existential sentence:  $\exists x_3 (\text{smokes}' x_3)$
- Since  $F_{\exists}$  and  $G_{\exists}$  should ‘correspond’, it follows that for meaning assignment  $h$ :  

$$h(\exists x_3 (\text{smokes}' x_3)) = h(F_{\exists}(x_3, (\text{smokes}' x_3))) = G_{\exists}(h(x_3), h((\text{smokes}' x_3)))$$
- Next, note that  $h((\text{smokes}' x_3)) \in M_{t, E, J}$  and for any  $g \in J$ ,  $h(x_3)(g) = f(x_3)(g) = g(x_3)$
- Thus:  $G_{\exists}(h(x_3), h((\text{smokes}' x_3))) =$   
The function  $E$  such that if  $g \in J$ ,  $E(g) = 1$  iff  
there is an  $x \in D_{e, E}$  such that  $h((\text{smokes}' x_3))(g(x_3/x)) = 1$
- Now, recall that for any  $g \in J$ ,  $h((\text{smokes}' x_3))(g) = s(g(x_3))$
- Thus:  $G_{\exists}(h(x_3), h((\text{smokes}' x_3))) =$   
The function  $E$  such that if  $g \in J$ ,  $E(g) = 1$  iff  
there is an  $x \in D_{e, E}$  such that  $s(g(x_3/x)(x_3)) = 1$
- Finally, by definition of ‘ $g(x_3/x)$ ’, we have it that:  

$$h(\exists x_3 (\text{smokes}' x_3)) = \text{The function } E \text{ such that if } g \in J, E(g) = 1 \text{ iff}$$
  
there is an  $x \in D_{e, E}$  such that  $s(x) = 1$

(46) **Remarks**

- Thus, the meaning of  $\exists x_3 (\text{smokes}' x_3)$  is a constant function on variable assignments
  - It maps any variable assignment  $g$  to 1 iff there is an  $x \in D_{e, E}$  such that the ‘denotation of **smokes**’ ( $s$ ) maps  $x$  to 1
- **Parallel with Models:**  
For any variable assignment  $g$ ,  $[[\exists x_3 (\text{smokes}' x_3)]]^{M,g} = 1$  iff there is an  $x \in D_{e, E}$  such that  $I(\text{smokes}')(x) = 1$ .

Given the discussion in (44)-(46), it can easily be seen that the following operations will similarly serve as semantic correlates of  $F_{\forall}$

(47) **The Operation  $G_{\forall}$**

- a. If there is a type  $\tau \in T$  and a variable  $v \in \text{VAR}_{\tau}$  such that for all  $g \in J$ ,  

$$\alpha(g) = g(v), \text{ and } \beta \in M_{t, E, J}, \text{ then}$$
  

$$G_{\forall}(\alpha, \beta) = \text{The function } A \text{ such that if } g \in J, A(g) = 1 \text{ iff}$$
  
for all  $x \in D_{\tau, E}$ ,  $\beta(g(v/x)) = 1$
- b. Otherwise,  $G_{\forall}(\alpha, \beta) = \text{garbage}$

And, last but not least, how do we handle the semantics of lambda abstraction?...

(48) **The Operation  $G_\lambda$**

- a. If there is a type  $\sigma \in T$  and a variable  $v \in \text{VAR}_\sigma$  such that for all  $g \in J$ ,  
 $\alpha(g) = g(v)$ , and  $\beta \in M_{t, E, J}$ , then  
 $G_\lambda(\alpha, \beta) = \text{The function } L \text{ such that if } g \in J, L(g) =$   
 $\text{The function } p \text{ with domain } D_{\sigma, E} \text{ such that for any } x \in D_{\sigma, E}, p(x) = \beta(g(v/x))$
- b. Otherwise,  $G_\lambda(\alpha, \beta) = \text{garbage}$

(49) **How Does This Work?**

- Again, the requirement that ‘for all  $g \in J, \alpha(g) = g(v)$ ’ ensures that if the first argument  $\alpha$  is *not* the meaning of a variable, then  $G_\lambda(\alpha, \beta) = \text{garbage}$ 
  - Thus, ‘( $\lambda$ mitt’ (smokes’ mitt’))’ will end up being interpreted as **garbage**
- Now, consider the simple formula:  $(\lambda x_3 (\text{smokes'} x_3))$
- Since  $F_\lambda$  and  $G_\lambda$  should ‘correspond’, it follows that for meaning assignment  $h$ :  
 $h( (\lambda x_3 (\text{smokes'} x_3)) ) = h( F_\lambda(x_3, (\text{smokes'} x_3)) ) = G_\lambda( h(x_3), h((\text{smokes'} x_3)) )$
- Next, note that  $h((\text{smokes'} x_3)) \in M_{t, E, J}$  and for any  $g \in J$ ,  $h(x_3)(g) = f(x_3)(g) = g(x_3)$
- Thus:  $G_\lambda( h(x_3), h((\text{smokes'} x_3)) ) =$   
 $\text{The function } L \text{ such that if } g \in J, L(g) =$   
 $\text{The function } p \text{ with domain } D_{e, E} \text{ such that for any } x \in D_{e, E},$   
 $p(x) = h((\text{smokes'} x_3))(g(x_3/x))$
- Now, recall that for any  $g \in J$ ,  $h((\text{smokes'} x_3))(g) = s(g(x_3))$
- Thus:  $G_\lambda( h(x_3), h((\text{smokes'} x_3)) ) =$   
 $\text{The function } L \text{ such that if } g \in J, L(g) =$   
 $\text{The function } p \text{ with domain } D_{e, E} \text{ such that for any } x \in D_{e, E},$   
 $p(x) = s(g(x_3/x)(x_3))$
- Finally, by definition of ‘ $g(x_3/x)$ ’, we have it that:  
 $h( (\lambda x_3 (\text{smokes'} x_3)) ) =$   
 $\text{The function } L \text{ such that if } g \in J, L(g) = \text{the function } p \text{ with domain } D_{e, E} \text{ such that}$   
 $\text{for any } x \in D_{e, E}, p(x) = s(x)$

(50) **Remarks**

- So, the meaning of  $(\lambda x_3(\text{smokes}^* x_3))$  is a constant function on variable assignments
  - It maps any variable assignment  $g$  to the function  $p$  which takes an  $x \in D_{e,E}$  and returns the value that the ‘denotation of smokes’ ( $s$ ) maps  $x$  to.
- **Parallel with Models:**  
For any variable assignment  $g$ ,  $[(\lambda x_3(\text{smokes}^* x_3))]^{M,g} =$   
The function  $p$  whose domain is  $D_{e,E}$  and whose range is  $D_{t,E}$ , and for all  $x \in D_{e,E}$   
 $p(x) = I(\text{smokes})(x)$ .

(51) **Summary**

It seems, then, that the operations  $\{G_\gamma\}_{\gamma \in \{\text{Concat, Not, And, Or, If, } \exists, \forall, \lambda\}}$  will serve nicely as ‘semantic correlates’ of our syntactic operations  $\{F_\gamma\}_{\gamma \in \{\text{Concat, Not, And, Or, If, } \exists, \forall, \lambda\}}$

*Consequently, let us wrap up our definition in (30) as follows...*

(52) **Logically Possible Partly-Fregean Interpretation of Politics+ $\lambda$**

A logically possible partly-Fregean interpretation of Politics+ $\lambda$  connected with  $E, J, \sigma$  is a partly-Fregean interpretation  $\mathbf{B} = \langle B, G_\gamma, f \rangle_{\gamma \in \{\text{Concat, Not, And, Or, If, } \exists, \forall, \lambda\}}$  for Politics+ $\lambda$  connected with  $E, J, \sigma$ , such that:

- a.  $J$  is the set of variable assignments based on  $E$ .
- b. If  $\varphi \in \text{CON}_\tau$ , then  $f(\varphi)(g) = f(\varphi)(g')$ , for all  $g, g' \in J$ .
- c. If  $\varphi \in \text{VAR}_\tau$ , then  $f(\varphi)(g) = g(\varphi)$  for all  $g \in J$ .
- d. The operations  $\{G_\gamma\}_{\gamma \in \{\text{Concat, Not, And, Or, If, } \exists, \forall, \lambda\}}$  are as defined in (31)-(48)

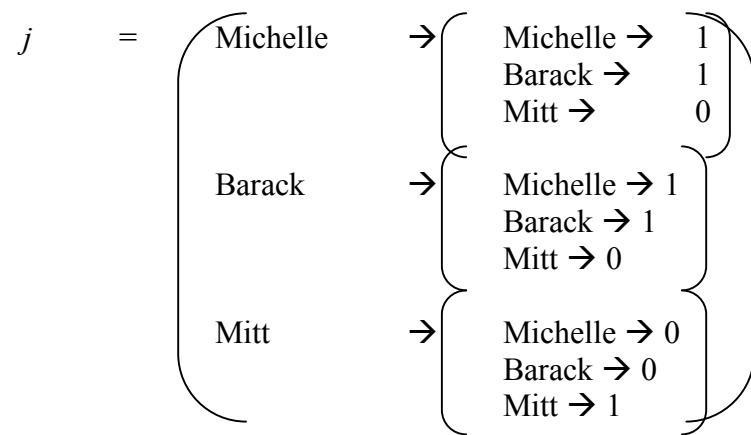
### 3. A Logically Possible Partly-Fregean Interpretation for Politics+ $\lambda$

Let us now illustrate the key notion in (52) by concretely spelling out such an interpretation.

#### (53) A Logically Possible Partly-Fregean Interpretation for Politics+ $\lambda$

Let E be the set {Barack, Michelle, Mitt}. Let J be the set of variable assignments based on E. Let  $\sigma$  be the type assignment for Politics+ $\lambda$  such that for all  $\tau \in T$ ,  $\sigma(\tau) = \sigma(<\text{var}, \tau>) = \tau$ . Let  $\mathbf{B} = \langle B, G_\gamma, f \rangle_{\gamma \in \{\text{Concat, Not, And, Or, If, } \exists, \forall, \lambda\}}$  be the logically possible partly-Fregean interpretation of Politics+ $\lambda$  connected with E, J,  $\sigma$  such that f consists of the following mappings.

- a.  $f(\text{barack}') =$  The function  $b$  such that for all  $g \in J$ ,  $b(g) = \text{Barack}$ .
- b.  $f(\text{michelle}') =$  The function  $m$  such that for all  $g \in J$ ,  $m(g) = \text{Michelle}$
- c.  $f(\text{mitt}') =$  The function  $mt$  such that for all  $g \in J$ ,  $mt(g) = \text{Mitt}$
- d.  $f(\text{smokes}') =$  The function  $s$  such that for all  $g \in J$ ,  $s(g) = o = \{ <\text{Michelle}, 0>, <\text{Barack}, 1>, <\text{Mitt}, 0> \}$
- e.  $f(\text{man}') =$  The function  $mn$  such that for all  $g \in J$ ,  $mn(g) = i = \{ <\text{Michelle}, 0>, <\text{Barack}, 1>, <\text{Mitt}, 1> \}$
- f.  $f(\text{president}') =$  The function  $pr$  such that for all  $g \in J$ ,  $pr(g) = k = \{ <\text{Michelle}, 0>, <\text{Barack}, 1>, <\text{Mitt}, 0> \}$
- g.  $f(\text{loves}') =$  The function  $l$  such that for all  $g \in J$ ,  $l(g) =$



Finally, as we've done so many times before, we can use the meaning assignment h determined by  $\mathbf{B}$  to assign meanings to the expressions of Politics+ $\lambda$

- To save space, I've not included all the calculation steps below. Students are encouraged to work out all the steps for themselves.

In the calculations below, let  $h$  be the meaning assignment determined by the interpretation  $\mathbf{B}$  defined in (53).

(54)  $\forall x_3 (\mathbf{man}' x_3)$

- $h(\forall x_3 (\mathbf{man}' x_3)) =$
- The function  $A$  such that if  $g \in J$ ,  $A(g) = 1$  iff for all  $x \in D_{e,E}$ ,  $i(x) = 1$   $=$
- The function  $A$  such that if  $g \in J$ ,  $A(g) = 0$

Compare:

Relative to the model  $\mathcal{M}$  defined in (13) on the previous handout, for any variable assignment  $g$ ,  $[[\forall x_3 (\mathbf{man}' x_3)]]^{M,g} = 0$

(55)  $\exists P_4 (P_4 \mathbf{michelle}')$

- $h(\exists P_4 (P_4 \mathbf{michelle}')) =$
- The function  $E$  such that if  $g \in J$ ,  $E(g) = 1$  iff  
there is an  $x \in D_{e,t>,E}$  such that  $x(\text{Michelle}) = 1$   $=$
- The function  $E$  such that if  $g \in J$ ,  $E(g) = 1$

Compare:

Relative to the model  $\mathcal{M}$  defined in (13) on the previous handout, for any variable assignment  $g$ ,  $[[\exists P_4 (P_4 \mathbf{michelle}')]]^{M,g} = 1$

(56)  $(\lambda x_3 ((\mathbf{man}' x_3) \& (\mathbf{smokes}' x_3)))$

- $h((\lambda x_3 ((\mathbf{man}' x_3) \& (\mathbf{smokes}' x_3)))) =$
- The function  $L$  such that if  $g \in J$ ,  $L(g) =$  the function  $p$  with domain  $D_{e,E}$  such that for any  $x \in D_{e,E}$ ,  $p(x) = 1$  iff  $i(x) = 1$  and  $o(x) = 1$ .  $=$
- The function  $L$  such that if  $g \in J$ ,  $L(g) = \{ <\text{Michelle}, 0>, <\text{Barack}, 1>, <\text{Mitt}, 0> \}$

Compare:

Relative to the model  $\mathcal{M}$  defined in (13) on the previous handout, for any variable assignment  $g$ ,  $[[ (\lambda x_3 ((\mathbf{man}' x_3) \& (\mathbf{smokes}' x_3)))]]^{M,g} =$   
 $\{ <\text{Michelle}, 0>, <\text{Barack}, 1>, <\text{Mitt}, 0> \}$

(57)  $(\lambda P_4 (P_4 \text{ mitt}'))$

- $h( (\lambda P_4 (P_4 \text{ mitt}')) ) =$
- The function  $L$  such that if  $g \in J$ ,  $L(g) =$  the function  $p$  with domain  $D_{e,t>E}$  such that for any  $x \in D_{e,t>E}$ ,  $p(x) = x(\text{Mitt})$

Compare:

Relative to the model  $\mathcal{M}$  defined in (13) on the previous handout, for any variable assignment  $g$ ,  $[[ (\lambda P_4 (P_4 \text{ mitt}')) ] ]^{M,g} =$

The function  $p$  with domain  $D_{e,t>E}$ , range  $D_{t,E}$  and for all  $x \in D_{e,t>E}$ ,  $p(x) = x(\text{Mitt})$

(58)  $(\lambda P_4 \forall x_3 ((\text{man}' x_3) \rightarrow (P_4 x_3)))$

- $h( (\lambda P_4 \forall x_3 ((\text{man}' x_3) \rightarrow (P_4 x_3))) ) =$
- The function  $L$  such that if  $g \in J$ ,  $L(g) =$  the function  $p$  with domain  $D_{e,t>E}$  such that for any  $x \in D_{e,t>E}$ ,  $p(x) = 1$  iff for all  $y \in D_{e,E}$ ,  $i(y) = 0$  or  $x(y) = 1$
- The function  $L$  such that if  $g \in J$ ,  $L(g) =$  the function  $p$  with domain  $D_{e,t>E}$  such that for any  $x \in D_{e,t>E}$ ,  $p(x) = 1$  iff for all  $y \in D_{e,E}$ , if  $i(y) = 1$  then  $x(y) = 1$

Compare:

Relative to the model  $\mathcal{M}$  defined in (13) on the previous handout, for any variable assignment  $g$ ,  $[[ (\lambda P_4 \forall x_3 ((\text{man}' x_3) \rightarrow (P_4 x_3))) ] ]^{M,g} =$

The function  $p$  with domain  $D_{e,t>E}$ , range  $D_{t,E}$  and for all  $x \in D_{e,t>E}$ ,  $p(x) = 1$  iff for all  $y \in D_{e,E}$ , if  $i(y) = 1$  then  $x(y) = 1$

(59)  $((\lambda x_3 ((\text{man}' x_3) \& (\text{smokes}' x_3))) \text{ mitt}')$

- $h( ((\lambda x_3 ((\text{man}' x_3) \& (\text{smokes}' x_3))) \text{ mitt}') ) =$
- The function  $F\text{-}A$  such that if  $g \in J$ , then  $F\text{-}A(g) = h((\lambda x_3 ((\text{man}' x_3) \& (\text{smokes}' x_3))))(g)( h(\text{mitt}')(g) ) =$  (by (53), (56))
- The function  $F\text{-}A$  such that if  $g \in J$ , then  $F\text{-}A(g) = L(g)( mt(g) ) =$  (by (53), (56))
- The function  $F\text{-}A$  such that if  $g \in J$ , then  $F\text{-}A(g) = \{ <\text{Michelle}, 0>, <\text{Barack}, 1>, <\text{Mitt}, 0> \}(\text{Mitt}) =$
- The function  $F\text{-}A$  such that if  $g \in J$ , then  $F\text{-}A(g) = 0$

Compare: Relative to the model  $\mathcal{M}$  defined in (13) on the previous handout, for any variable assignment  $g$ ,  $[[ ((\lambda x_3 ((\text{man}' x_3) \& (\text{smokes}' x_3))) \text{ mitt}') ] ]^{M,g} = 0$

#### 4. Relationship Between Models and Logically Possible Partly-Fregean Interpretation

We now have two ways of providing Politics+ $\lambda$  a formal semantics:

- (i) The model-theoretic semantics from the handout ‘Preliminaries’
- (ii) The logically possible partly-Fregean interpretation in (53)

(60) **Fun Fact:** There is (again) an important equivalence between these two systems!

#### (61) The Relationship Between Models and Interpretations (Part 1)

Let  $\mathcal{M}$  be a model  $\langle E, I \rangle$  for Politics+ $\lambda$ . Let  $J$  be the set of variable assignments based on  $\mathcal{M}$ . Let  $B = \bigcup_{\tau \in T} M_{\tau, E, J}$ .

Finally, let  $f$  be the function whose domain is  $\bigcup_{\delta \in \Delta} X_\delta$  and for any  $\varphi \in \bigcup_{\delta \in \Delta} X_\delta$ ,  $f(\varphi) =$  the function  $m$  whose domain is  $J$  and for any  $g \in J$ ,  $m(g) = [[\varphi]]^{M, g}$

- a. Claim 1:

The structure  $\mathbf{B} = \langle B, G_\gamma, f \rangle_{\gamma \in \{\text{Concat, Not, And, Or, If, } \exists, \forall, \lambda\}}$  is a logically-possible partly-Fregean interpretation of Politics+ $\lambda$ .

- b. Claim 2:

Let  $h$  be the meaning assignment determined by  $\mathbf{B}$ . Every meaningful expression  $\varphi$  of Politics+ $\lambda$  is such that, for any variable assignment  $g \in J$ :

$$h(\varphi)(g) = [[\varphi]]^{M, g}$$

#### (62) Remark

Thus, if we are given a model  $\mathcal{M}$  for Politics+ $\lambda$ , we can construct a logically possible partly-Fregean interpretation  $\mathbf{B}$  that assigns equivalent values to the meaningful expressions of Politics+ $\lambda$ .

#### (63) The Relationship Between Models and Interpretations (Part 1)

Let  $\mathbf{B} = \langle B, G_\gamma, f \rangle_{\gamma \in \{\text{Concat, Not, And, Or, If, } \exists, \forall, \lambda\}}$  be a logically possible partly-Fregean interpretation of Politics+ $\lambda$  connected with  $E, J, \sigma$ . Let  $I$  be the function whose domain is  $\bigcup_{\tau \in T} CON_\tau$  and for any  $\varphi \in \bigcup_{\tau \in T} CON_\tau$ ,  $I(\varphi) = f(\varphi)(g)$ , for an arbitrary  $g$ .

- a. Claim 1: The structure  $\mathcal{M} = \langle E, I \rangle$  is a model of Politics+ $\lambda$ .

- b. Claim 2:

Let  $h$  be the meaning assignment determined by  $\mathbf{B}$ . Every meaningful expression  $\varphi$  of Politics+ $\lambda$  is such that, for any variable assignment  $g \in J$ :

$$h(\varphi)(g) = [[\varphi]]^{M, g}$$

(63) **Remark**

Thus, if we are given a logically possible partly-Fregean interpretation **B** of Politics+ $\lambda$ , we can construct a model  $\mathcal{M}$  for Politics+ $\lambda$  that assigns equivalent values to the meaningful expressions of Politics+ $\lambda$ .

(64) **Summary**

- Given this relationship between models and logically possible partly-Fregean interpretations of Politics+ $\lambda$ , we can freely shift between the two (*cf.* sets and their characteristic functions).
- Similarly, we can view these two systems as being *in essence* ‘the same thing’ (even though they are *very* different set theoretic objects).
- **Thus, even when we’re presenting our semantics in model-theoretic terms, we are also thereby providing a (Montagovian) interpretation for the language.**

(65) **The Importance of This Point**

- As you can probably tell now, once we’ve got variable binding in the language, the model-theoretic presentation of the semantics is *much simpler and more comprehensible* than its presentation as a (Montagovian) ‘interpretation.’
- **The difference in comprehensibility is even more profound for Montague’s system of ‘Intensional Logic’, used in PTQ.**
- For this reason, in PTQ (but not UG), Montague presents the semantics for IL in strictly model-theoretic terms (no ‘interpretations’).
  - I will be following suit, **but do know that there is straight-forward way of converting Montague’s model structures for IL into ‘interpretations’...**  
(just see UG, if you’re interested!...)

### Computing with ‘Logically Possible Partly-Fregean Interpretations’

Let  $h$  be the meaning assignment determined by the logically possible partly-Fregean interpretation in (53) of the handout ‘Fregean Interpretations.’”

- (i)  $h((\lambda P_4 \exists x_3 ((\text{man}' x_3) \& (P_4 x_3)))) =$  (by definition of Politics+ $\lambda$ )
- (ii)  $h(F_\lambda(P_4, F_\exists(x_3, F_{\text{And}}(F_{\text{Concat}}(\text{man}, x_3), F_{\text{Concat}}(P_4, x_3)\dots))) =$  (by homomorph. of  $h$ )
- (iii)  $G_\lambda(h(P_4), G_\exists(h(x_3), G_{\text{And}}(G_{\text{Concat}}(h(\text{man}), h(x_3)), G_{\text{Concat}}(h(P_4), h(x_3))\dots)) =$  (by def. of  $h$ )
- (iv)  $G_\lambda(f(P_4), G_\exists(f(x_3), G_{\text{And}}(G_{\text{Concat}}(f(\text{man}), f(x_3)), G_{\text{Concat}}(f(P_4), f(x_3))\dots)) =$  (by def. of  $f$ )
- (v)  $G_\lambda(f(P_4), G_\exists(f(x_3), G_{\text{And}}(G_{\text{Concat}}(mn, f(x_3)), G_{\text{Concat}}(f(P_4), f(x_3))\dots)) =$  (by def. of  $G_\lambda$ )
- (vi) The function  $L$  such that if  $g \in J, L(g) =$   
The function  $p$  with domain  $D_{e,t,E}$  such that for any  $x \in D_{e,t,E}$ ,  

$$p(x) = G_\exists(f(x_3), G_{\text{And}}(G_{\text{Concat}}(mn, f(x_3)), G_{\text{Concat}}(f(P_4), f(x_3))\dots)(g(P_4/x)) =$$
 (by def. of  $G_\exists$ )
- (vii) The function  $L$  such that if  $g \in J, L(g) =$   
The function  $p$  with domain  $D_{e,t,E}$  such that for any  $x \in D_{e,t,E}$ ,  

$$p(x) = [\text{The function } E \text{ such that if } g' \in J, E(g') = 1 \text{ iff there is a } y \in D_{e,E} \text{ such that } G_{\text{And}}(G_{\text{Concat}}(mn, f(x_3)), G_{\text{Concat}}(f(P_4), f(x_3))\dots)(g'(x_3/y)) = 1](g(P_4/x)) =$$
 (by meta-logic)
- (viii) The function  $L$  such that if  $g \in J, L(g) =$   
The function  $p$  with domain  $D_{e,t,E}$  such that for any  $x \in D_{e,t,E}$ ,  

$$p(x) = 1 \text{ iff there is an } y \in D_{e,E} \text{ such that } G_{\text{And}}(G_{\text{Concat}}(mn, f(x_3)), G_{\text{Concat}}(f(P_4), f(x_3))\dots)(g(P_4/x)(x_3/y)) = 1 =$$
 (by def. of  $G_{\text{And}}$ )
- (ix) The function  $L$  such that if  $g \in J, L(g) =$   
The function  $p$  with domain  $D_{e,t,E}$  such that for any  $x \in D_{e,t,E}$ ,  

$$p(x) = 1 \text{ iff there is an } y \in D_{e,E} \text{ such that } [\text{The function } Conj \text{ such that if } g' \in J, \text{ then } Conj(g') = 1 \text{ iff } G_{\text{Concat}}(mn, f(x_3))(g') = 1 \text{ and } G_{\text{Concat}}(f(P_4), f(x_3))(g') = 1](g(P_4/x)(x_3/y)) = 1 =$$
 (by meta-logic)

- (x) The function  $L$  such that if  $g \in J$ ,  $L(g) =$   
 The function  $p$  with domain  $D_{\langle e,t \rangle, E}$  such that for any  $x \in D_{\langle e,t \rangle, E}$ ,  
 $p(x) = 1$  iff there is an  $y \in D_{e,E}$  such that  
 $G_{Concat}( mn , f(x_3) )(g(P_4/x)(x_3/y)) = 1$  and  $G_{Concat}( f(P_4), f(x_3))(g(P_4/x)(x_3/y)) = 1$   
 $=$  (by def. of  $G_{Concat}$ )
- (xi) The function  $L$  such that if  $g \in J$ ,  $L(g) =$   
 The function  $p$  with domain  $D_{\langle e,t \rangle, E}$  such that for any  $x \in D_{\langle e,t \rangle, E}$ ,  
 $p(x) = 1$  iff there is an  $y \in D_{e,E}$  such that  
 $mn((g(P_4/x)(x_3/y))(f(x_3)(g(P_4/x)(x_3/y)))) = 1$  and  
 $f(P_4)((g(P_4/x)(x_3/y))(f(x_3)(g(P_4/x)(x_3/y)))) = 1$   
 $=$  (by (53))
- (xii) The function  $L$  such that if  $g \in J$ ,  $L(g) =$   
 The function  $p$  with domain  $D_{\langle e,t \rangle, E}$  such that for any  $x \in D_{\langle e,t \rangle, E}$ ,  
 $p(x) = 1$  iff there is an  $y \in D_{e,E}$  such that  
 $i(g(P_4/x)(x_3/y))(x_3) = 1$  and  $g(P_4/x)(x_3/y)(P_4)(g(P_4/x)(x_3/y))(x_3) = 1$   
 $=$  (by def. of ‘Logically Possible Partly-Fregean Interpretation’)
- (xiii) The function  $L$  such that if  $g \in J$ ,  $L(g) =$   
 The function  $p$  with domain  $D_{\langle e,t \rangle, E}$  such that for any  $x \in D_{\langle e,t \rangle, E}$ ,  
 $p(x) = 1$  iff there is an  $y \in D_{e,E}$  such that  $i(y) = 1$  and  $x(y) = 1$

Thus,  $h((\lambda P_4 \exists x_3 ((\text{man}' x_3) \& (P_4 x_3))))$  is computed to be a constant function on variable assignments.

- It maps every variable assignment to the function  $p$ , which is the characteristic function of the set of  $\langle e,t \rangle$  functions  $f$  which map some man to 1.
- Thus, it maps every variable assignment to the ‘set of properties that some man has’

Note the parallel to our model theoretic semantics from the handout “Preliminaries”:

Let  $\mathcal{M}$  be the model defined in (13) of “Preliminaries”. Let  $g$  be any variable assignment based on  $\mathcal{M}$ .

$$[[ (\lambda P_4 \exists x_3 ((\text{man}' x_3) \& (P_4 x_3))) ]]^{\mathcal{M}, g} =$$

The function  $p$  with domain  $D_{\langle e,t \rangle, E}$ , range  $D_{t,E}$  and for all  $x \in D_{\langle e,t \rangle, E}$ ,  
 $p(x) = 1$  iff there is a  $y \in D_{e,E}$  such that  $i(y) = 1$  and  $x(y) = 1$

## An Algebraic Approach to Quantification and Lambda Abstraction: Applications to the Analysis of English

- (1) **Ingredients on the Table**
- A logical language with both quantification and lambda abstraction (Politics+ $\lambda$ )
  - An interpretation  $\langle B, G_\gamma, f \rangle_{\gamma \in \{\text{Concat, Not, And, Or, If, } \exists, \forall, \lambda\}}$  for Politics+ $\lambda$

(2) **Goals for This Unit**

- Develop a fragment of English that contains quantificational NPs.
- Develop a translation base from that fragment to Politics+ $\lambda$
- Examine the interpretation thereby induced for the fragment of English.

**Note:** The system developed here is a blend of the ones Montague develops in UG and PTQ. To my knowledge, this system appears nowhere else in the literature on MG.

---

### 1. Towards a New Fragment of English: Quantificational NPs in Subject Position

*In this section, we will work out at a relatively informal level how we should structure the English fragment and its translation into Politics+ $\lambda$*

(3) **Proximal Goal**

Expand our fragment of English so it includes expressions like: *some man, every man*.

- (4) a. Easy Question: What should the category of these new expressions be?
- b. Easy Answer:  
Well, their external syntactic behavior is just like the expressions *Barack, Mitt, Michelle*. Thus, if the latter are NPs, then so should be *some man* and *every man*.

(5) **First Steps Towards Implementation**

We're not ready to officially write out this new fragment of English, but it seems that it should include the following pieces.

- The New Category Labels:
  - TV ‘transitive verb’
  - IV ‘intransitive verb’ (and ‘VPs’)
  - S ‘sentence’
  - T ‘term’ (formerly ‘NP’)**
  - CN ‘common noun’ (that is, the ‘Ns’)

b. The Basic Expressions

- (i)  $X_{TV} = \{\langle loves, \emptyset \rangle\}$
- (ii)  $X_{IV} = \{\langle smokes, \emptyset \rangle\}$
- (iii)  $X_T = \{\langle Barack, \emptyset \rangle, \langle Mitt, \emptyset \rangle, \langle Michelle, \emptyset \rangle\}$
- (iv)  $X_{CN} = \{\langle man, \emptyset \rangle, \langle president, \emptyset \rangle\}$
- (v)  $X_S = \emptyset$

c. The Syntactic Operations

In addition to the operations we introduced for DME, let's add the operations  $K_{Some}$  and  $K_{Every}$ .

- In the definitions below,  $\alpha$  and  $\beta$  are trees whose root nodes are ordered pairs.  
In addition  $\alpha'$  and  $\beta'$  are the first members of the root nodes of  $\alpha$  and  $\beta$ .

$$(i) K_{Some}(\alpha) = \langle \text{some } \alpha', \text{Some} \rangle$$

$\alpha$

$$(ii) K_{Every}(\alpha) = \langle \text{every } \alpha', \text{Every} \rangle$$

$\alpha$

d. The Syntactic Rules

Let's add the following rules to the set of syntactic rules  $S_E$  we had for DME.

$$(i) \langle K_{Some}, \langle CN \rangle, T \rangle$$

'The result of applying  $K_{Some}$  to an expression of category CN is an expression of category T.'

$$(ii) \langle K_{Every}, \langle CN \rangle, T \rangle$$

'The result of applying  $K_{Every}$  to an expression of category CN is an expression of category T.'

e. Illustration: The category  $C_T$  includes the following trees:

$$\langle Barack, \emptyset \rangle \quad \langle Mitt, \emptyset \rangle \quad \langle Michelle, \emptyset \rangle$$

$$\begin{array}{ccc} \langle \text{every man, Every} \rangle & & \langle \text{every president, Every} \rangle \\ | & & | \\ \langle man, \emptyset \rangle & & \langle president, \emptyset \rangle \end{array}$$

$$\begin{array}{ccc} \langle \text{some man, Some} \rangle & & \langle \text{some president, Some} \rangle \\ | & & | \\ \langle man, \emptyset \rangle & & \langle president, \emptyset \rangle \end{array}$$

(6) **Handy Abbreviation**

Since it's clear from context here that the expressions of our fragment are *trees*, I'll make use of the convention below to save space:

$$\underline{\text{expression}} \quad = \quad \text{A tree whose root node is } <\text{expression}, \text{Index}>$$

(7) **Difficult Question:**

What type of expressions in Politics+ $\lambda$  should our (quantificational) terms translate as?

(8) **Towards the Answer (Linguistics 610)**

- We don't want the meaning of some man or every man end up being of type e
- **Rather, we want it to be a function of type  $\langle\langle e, t \rangle, t \rangle$  (Generalized Quantifier)**

$$h^o k(\underline{\text{some man}}) \quad = \quad$$

the function whose domain is  $D_{\langle et \rangle, E}$ , and for every  $f$  in  $D_{\langle et \rangle, E}$  maps  $f$  to 1 iff there is an  $x$  in  $D_{e, E}$  such that  $x$  is a man and  $f(x) = 1$

$$h^o k(\underline{\text{every man}}) \quad = \quad$$

the function whose domain is  $D_{\langle et \rangle, E}$ , and for every  $f$  in  $D_{\langle et \rangle, E}$  maps  $f$  to 1 iff for all  $x$  in  $D_{e, E}$ , if  $x$  is a man then  $f(x) = 1$

- Therefore, we'd like our translation function  $k$  to achieve the following mappings:

$$k(\underline{\text{some man}}) \quad = \quad (\lambda P_0 \exists x_0 ((\text{man}' x_0) \& (P_0 x_0)))$$

$$k(\underline{\text{every man}}) \quad = \quad (\lambda P_0 \forall x_0 ((\text{man}' x_0) \rightarrow (P_0 x_0)))$$

(9) **A Challenge Appears!**

- Thus, we want our translation function  $k$  to map some man and every man to expressions in Politics+ $\lambda$  of type  $\langle\langle e, t \rangle, t \rangle$
- But, recall that under our theory of translation, every expression of the same category in English must be translated to an expression of the same type in Politics+ $\lambda$
- **Thus, the goal in (8) entails that *every term of our language – even Barack, Michelle, and Mitt – must be translated as  $\langle\langle et \rangle, t \rangle$  formulae.***

(10) **The Solution to the Challenge (Linguistics 610)**

The proper names Barack, Michelle, and Mitt will receive the translations below:

- a.  $k(<\text{Barack}, \emptyset>) = (\lambda P_0 (P_0 \text{barack}'))$
- b.  $k(<\text{Mitt}, \emptyset>) = (\lambda P_0 (P_0 \text{mitt}'))$
- c.  $k(<\text{Michelle}, \emptyset>) = (\lambda P_0 (P_0 \text{michelle}'))$

That is, the name (e.g.) Barack will end up translated as an expression denoting the characteristic function of the set of  $\langle e, t \rangle$  functions that map Barack to 1.

(11) **Immediate Problem**

In our translation base mapping Mini-English to Politics-NoQ, the syntactic operation  $K_{\text{Merge-S}}$  corresponds to the polynomial operation  $F_{\text{Concat}} < \text{Id}_{2,2}, \text{Id}_{1,1} >$ .

- Given the translations in (8)-(10), if we maintain that correspondence, we'll end up mapping meaningful expressions of Mini-English to syntactic garbage in Politics+ $\lambda$ .

$$\begin{aligned}
 k(\underline{\text{Barack smokes}}) &= \\
 k(K_{\text{Merge-S}}(<\text{Barack}, \emptyset>, <\text{smokes}, \emptyset>)) &= \\
 F_{\text{Concat}} < \text{Id}_{2,2}, \text{Id}_{1,1} > (k(<\text{Barack}, \emptyset>), k(<\text{smokes}, \emptyset>)) &= \\
 F_{\text{Concat}} < \text{Id}_{2,2}, \text{Id}_{1,1} > ( (\lambda P_0 (P_0 \text{barack}')), \text{smokes}') &= \\
 (\text{smokes}' (\lambda P_0 (P_0 \text{barack}'))) &= \text{syntactic garbage!}
 \end{aligned}$$

(12) **Potential Solution**

For our new fragment of English, in the translation to Politics+ $\lambda$ ,  $K_{\text{Merge-S}}$  could now just correspond to  $F_{\text{Concat}}$ . That would yield the right results!

$$\begin{aligned}
 k(\underline{\text{Barack smokes}}) &= \\
 k(K_{\text{Merge-S}}(<\text{Barack}, \emptyset>, <\text{smokes}, \emptyset>)) &= \\
 F_{\text{Concat}}(k(<\text{Barack}, \emptyset>), k(<\text{smokes}, \emptyset>)) &= \text{(by (10))} \\
 F_{\text{Concat}}( (\lambda P_0 (P_0 \text{barack}')), \text{smokes}') &= \\
 ((\lambda P_0 (P_0 \text{barack}')) \text{smokes}') &\Leftrightarrow \text{(by lambda-conversion)} \\
 (\text{smokes}' \text{barack}') &
 \end{aligned}$$

*Our solution in (12) would also get the right results for sentences whose subjects are quantificational terms!...*

(13) **Illustration: Every man smokes**

$$\begin{aligned}
 k(\underline{\text{every man smokes}}) &= \\
 k(K_{\text{Merge-S}}(\underline{\text{every man}}, \langle \text{smokes}, \emptyset \rangle)) &= \\
 F_{\text{Concat}}(k(\underline{\text{every man}}), k(\langle \text{smokes}, \emptyset \rangle)) &= \quad (\text{by (8)}) \\
 F_{\text{Concat}}((\lambda P_0 \forall x_0 ((\text{man}' x_0) \rightarrow (P_0 x_0))), \text{smokes}') &= \\
 ((\lambda P_0 \forall x_0 ((\text{man}' x_0) \rightarrow (P_0 x_0))) \text{smokes}') &\Leftrightarrow \quad (\text{by lambda-conversion}) \\
 \forall x_0 ((\text{man}' x_0) \rightarrow (\text{smokes}' x_0))
 \end{aligned}$$

(14) **Illustration: Some man smokes**

$$\begin{aligned}
 k(\underline{\text{some man smokes}}) &= \\
 k(K_{\text{Merge-S}}(\underline{\text{some man}}, \langle \text{smokes}, \emptyset \rangle)) &= \\
 F_{\text{Concat}}(k(\underline{\text{some man}}), k(\langle \text{smokes}, \emptyset \rangle)) &= \quad (\text{by (8)}) \\
 F_{\text{Concat}}((\lambda P_0 \exists x_0 ((\text{man}' x_0) \& (P_0 x_0))), \text{smokes}') &= \\
 ((\lambda P_0 \exists x_0 ((\text{man}' x_0) \& (P_0 x_0))) \text{smokes}') &\Leftrightarrow \quad (\text{by lambda-conversion}) \\
 \exists x_0 ((\text{man}' x_0) \& (\text{smokes}' x_0))
 \end{aligned}$$

(15) **Summary**

Well, it looks like we're all done here! We just need to properly implement the ideas in (5)-(12), and we can call it a day, put on our PJs, and get caught up on Breaking Bad!...

## 2. Towards a New Fragment of English: Quantificational NPs in Object Position

*Oh wait... Crap! We forgot about the fact that quantificational terms can also be direct objects...*

(16) **New Goal**

We would like for the sentences below to paired with the following translations:

- a. Sentence: *Michelle loves every president.*  
Translation:  $\forall x_0 ((\text{president}' x_0) \rightarrow ((\text{loves}' x_0) \text{ michelle}'))$
- b. Sentence: *Mitt loves some man.*  
Translation:  $\exists x_0 ((\text{man}' x_0) \& ((\text{loves}' x_0) \text{ mitt}'))$

(17) **The Obvious Problem**

In our translation base mapping Mini-English to Politics-NoQ, the syntactic operation  $K_{\text{Merge-IV}}$  corresponds to the polynomial operation  $F_{\text{Concat}}$ .

- Given the translations in (8)-(10), if we maintain that correspondence, we'll end up mapping meaningful expressions of Mini-English to syntactic garbage in Politics+ $\lambda$ .
  - And, we definitely don't get translations like (16a,b)

$$\begin{aligned}
 k(\underline{\text{loves every president}}) &= \\
 k(K_{\text{Merge-IV}}(<\text{loves}, \emptyset>, \underline{\text{every president}})) &= \\
 F_{\text{Concat}}(k(<\text{loves}, \emptyset>), k(\underline{\text{every president}})) &= \quad (\text{by (8)}) \\
 F_{\text{Concat}}(\text{loves}', (\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))) ) &= \\
 (\text{loves}' (\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))) ) &= \quad \text{syntactic garbage!}
 \end{aligned}$$

(18) **The More General Issue: Quantificational Terms in Object Position**

- In the semantics for English that we get from our translation, we want terms like every president to end up denoting functions of type  $\langle e, t \rangle, t \rangle$  (GQs)
- But we also in our semantics want transitive verbs like smokes to end up denoting functions of type  $\langle e, \langle e, t \rangle, t \rangle$  (curried characteristic functions of binary relations).
- But, how do we put together meanings of those two different types?**

(19) **The PTQ Solution, Rough Idea: Part 1**

Our fragment of English will contain a syntactic operation that is like 'QR-in-reverse'.

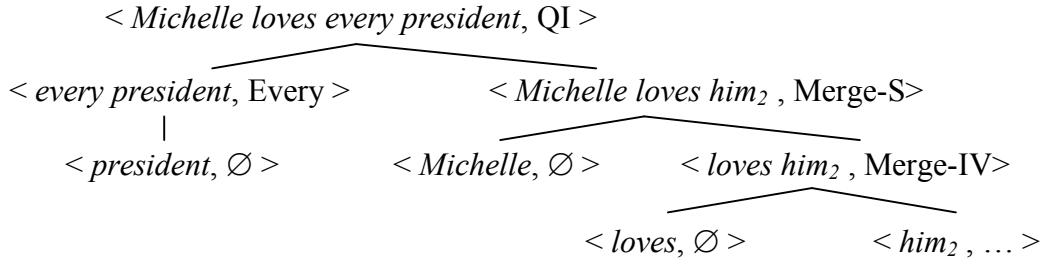
- This operation  $K_{\text{QI}}$  (QI = 'quantifying in') will take (i) a quantificational term, and (ii) a sentence containing a pronoun (*cf.* trace), and output a string where the pronoun (*cf.* trace) is **replaced** with the quantification term.

Rough Idea Illustrated:

$$K_{\text{QI}}(\underline{\text{every president}}, \underline{\text{Michelle loves him}_2}) = \underline{\text{Michelle loves every president.}}$$

(20) **Remark**

Although the transformation of the strings is like ‘QR-in-reverse’, the actual analysis trees output by  $K_{QI}$  are geometrically similar to the PS-trees generated by QR.



(21) **The PTQ Solution, Rough Idea: Part 2**

The polynomial operation corresponding to  $K_{QI}$  ( $H_{QI}$ ) will do all the following:

- (i) Take the translation of the quantificational term
- (ii) Take the translation of the sentence
- (iii) Lambda-abstract over the variable in the translation of the sentence
- (iv) Apply the translation of the quantificational term to the resulting  $\lambda$ -expression

<div style="border: 1px solid black; padding: 5px; display: inline-block;">The Key Step!</div>	$k(K_{QI}(\underline{\text{every president}}, \underline{\text{Michelle loves him}_2})) =$ $H_{QI}(k(\underline{\text{every president}}), k(\underline{\text{Michelle loves him}}_2)) =$ $\left\{ \begin{array}{l} H_{QI}((\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))), ((\text{loves}' x_0) \text{ michelle}')) \\ ((\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))) (\lambda x_0 ((\text{loves}' x_0) \text{ michelle}'))) \end{array} \right. \begin{array}{l} = \\ \Leftrightarrow \\ (\text{by alpha-conversion}) \end{array}$ $((\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))) (\lambda x_1 ((\text{loves}' x_1) \text{ michelle}'))) \begin{array}{l} \Leftrightarrow \\ (\text{by lambda-conversion}) \end{array}$ $\forall x_0 ((\text{president}' x_0) \rightarrow ((\lambda x_1 ((\text{loves}' x_1) \text{ michelle}')) x_0)) \begin{array}{l} \Leftrightarrow \\ (\text{by lambda-conversion}) \end{array}$ $\forall x_0 ((\text{president}' x_0) \rightarrow ((\text{loves}' x_0) \text{ michelle}'))$
--	--

If we can flesh out the ideas in (19) and (21), we will have a system that properly translates / interprets quantificational terms in object position.

(22) **Our To-Do List**

- a. Step One: Develop a theory of the syntax and semantics of pronouns in English
- b. Step Two: Properly define the operation  $K_{QI}$
- c. Step Three: Properly define the operation  $H_{QI}$

(23) **Fun Fact!**

Once we've set up everything correctly, our system will correctly predict quantifier-scope ambiguities. That is, our system will predict that the English sentence in (a) can receive either of the translations in (b).

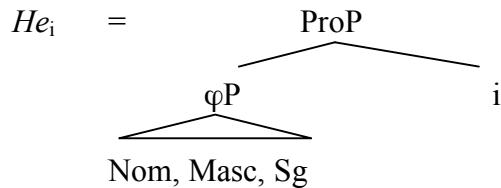
- a. Sentence of (Ambiguous) English: *Some man loves every president.*
- b. Possible Translations into Politics+ $\lambda$ 
  - (i)  $\exists x_0 ((\text{man}' x_0) \& \forall x_1 ((\text{president}' x_1) \rightarrow ((\text{loves}' x_1) x_0)))$
  - (ii)  $\forall x_0 ((\text{president}' x_0) \rightarrow \exists x_1 ((\text{man}' x_1) \& ((\text{loves}' x_0) x_1)))$

### 3. Developing The Analysis of English Quantification

#### 3.1 The Syntax and Translation of English Pronouns

(24) **A Radical, Crazy Idea Only Chomskians Could Come Up With**

Perhaps pronouns are structurally complex, and consist of (i) a 'pronominal core' containing so-called 'φ-features' like Case, Gender, Number, and (ii) a pronominal index.



(25) **Montagovian Precursor (in "Universal Grammar")**

Pronouns in English are actually syntactically derived from more primitive 'indices'.

$$K_{He}(i) = \underline{He}_i$$

(26) **Adapting This Idea to Our System: The Syntax of Indices**

We will assume that indices are trivial trees consisting of the following root-nodes:

$$\text{IND} = \{ < n, \emptyset > : n \in \mathbb{N} \}$$

(27) **The Translation of Pronouns into Politics+ $\lambda$**

The system we will go on to build will not translate the pronouns *per se*, but rather the indices the pronouns are derived from.

- a. Translation of Indices: For all  $n \in \mathbb{N}$ ,  $k(<n, \emptyset>) = x_n (= v_{e,n})$
- b. Polynomial Operation for  $K_{He}$ :  $H_{He} = Id_{1,1}$
- c. Illustration:  $k(\underline{he}\ 3) = k(K_{He}(<3, \emptyset>)) = Id_{1,1}(x_3) = x_3$   
 $H_{He}(k(<3, \emptyset>)) = (by\ (27a),\ (27b))$

(28) **Partial Implementation of the Ideas in (25)-(27)**

- a. Expanding Our Set of Category Labels for English:

- (i) TV ‘transitive verb’
- (ii) IV ‘intransitive verb’ (and ‘VPs’)
- (iii) S ‘sentence’
- (iv) T ‘term’ (formerly ‘NP’)
- (v) CN ‘common noun’ (that is, the ‘Ns’)
- (vi) IN ‘indices’
- (vii) PR ‘pronouns’

- b. The Basic Expressions:

- (i)  $X_{TV} = \{ <\text{loves}, \emptyset> \}$
- (ii)  $X_{IV} = \{ <\text{smokes}, \emptyset> \}$
- (iii)  $X_T = \{ <\text{Barack}, \emptyset>, <\text{Mitt}, \emptyset>, <\text{Michelle}, \emptyset> \}$
- (iv)  $X_{CN} = \{ <\text{man}, \emptyset>, <\text{president}, \emptyset> \}$
- (v)  $X_{IN} = \{ <n, \emptyset> : n \in \mathbb{N} \}$
- (vi)  $X_{PR} = \emptyset$
- (vii)  $X_S = \emptyset$

- c. The Syntactic Operations:

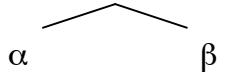
In addition to the operations we already have, let's add the operations  $K_{He}$  and  $K_{She}$ . We'll also revise the operation  $K_{Merge-IV}$  as in (iii) below:

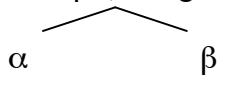
- In the definitions below,  $\alpha$  and  $\beta$  are trees whose root nodes are ordered pairs.  
 In addition  $\alpha'$  and  $\beta'$  are the first members of the root nodes of  $\alpha$  and  $\beta$ .

- (i)  $K_{He}(\alpha) = <\text{he } \alpha', \text{He}>$   

$$\begin{array}{c} | \\ \alpha \end{array}$$
- (ii)  $K_{She}(\alpha) = <\text{she } \alpha', \text{She}>$   

$$\begin{array}{c} | \\ \alpha \end{array}$$

- (iii)  $K_{\text{Merge-IV}}(\alpha, \beta) =$
1. If  $\beta' = \text{he } n$ , then  $\langle \alpha' \text{ him } n, \text{Merge-IV} \rangle$   

  2. If  $\beta' = \text{she } n$ , then  $\langle \alpha' \text{ her } n, \text{Merge-IV} \rangle$   

  3. Otherwise  $\langle \alpha' \beta', \text{Merge-IV} \rangle$   


Note: To capture the effects of pronominal case, we now have independent, empirical motivation for distinguishing the operations  $K_{\text{Merge-IV}}$  and  $K_{\text{Merge-S}}$

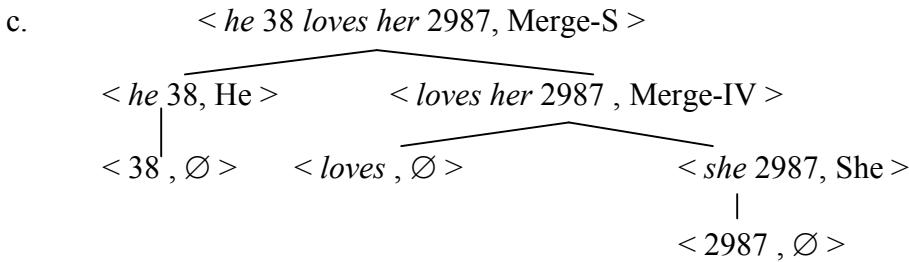
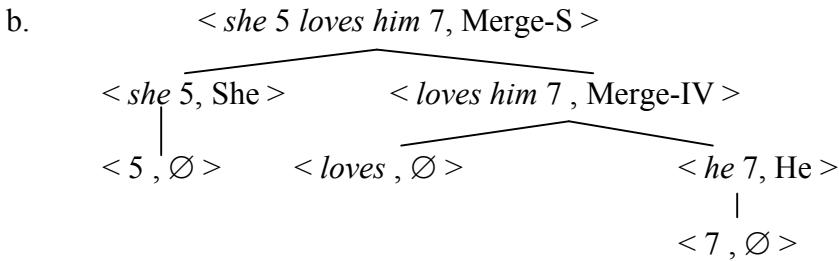
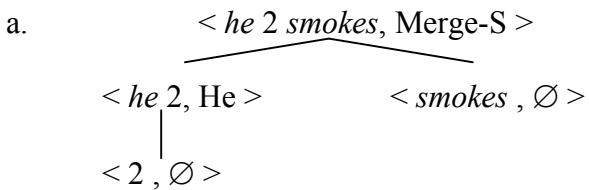
d. The Syntactic Rules:

Let us remove from our fragment of English the rules  $\langle K_{\text{Merge-IV}}, \langle \text{TV}, \text{NP} \rangle, \text{IV} \rangle$  and  $\langle K_{\text{Merge-S}}, \langle \text{NP}, \text{IV} \rangle, \text{S} \rangle$ . Let us add the following rules:

- (i)  $\langle K_{\text{He}}, \langle \text{IN} \rangle, \text{PR} \rangle$   
 ‘Applying  $K_{\text{He}}$  to an expression of category IN (an index) yields an expression of category PR (a pronoun).’
- (ii)  $\langle K_{\text{She}}, \langle \text{IN} \rangle, \text{PR} \rangle$   
 ‘Applying  $K_{\text{She}}$  to an expression of category IN (an index) yields an expression of category PR (a pronoun).’
- (iii)  $\langle K_{\text{Merge-IV}}, \langle \text{TV}, \text{PR} \rangle, \text{IV} \rangle$   
 ‘Applying  $K_{\text{Merge-IV}}$  to an expression of category TV (transitive verb) and one of category PR (pronoun) yields an expression of category IV.’
- (iv)  $\langle K_{\text{Merge-S}}, \langle \text{PR}, \text{IV} \rangle, \text{S} \rangle$   
 ‘Applying  $K_{\text{Merge-S}}$  to an expression of category PR (pronoun) and one of category IV yields an expression of category S.’

(29) **Illustration of the Resulting System**

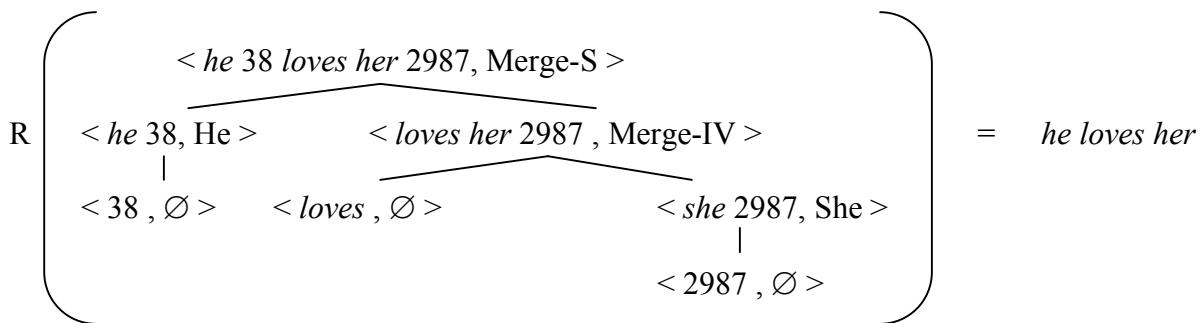
The system sketched in (28) will entail that  $C_S$  contains trees such as:



Since we don't actually pronounce indices in English, we will want our 'ambiguating' relation  $R$  to delete them from the first member of the root-node of our trees.

(30) **A Slight Change to Our 'Ambiguating' Relation  $R$**

$R$  is a function which takes as input a tree  $T$ , and returns as output the first member of the root node of  $T$  without any numerals.



(31) **A Sketch of the Translation to Politics+λ**

The correspondences in (31a), added with (27a), will yield the translations in (31b).

a. Correspondences Between Syntactic Operations and Polynomial Operations:

(i)	$K_{He}$	$\sim$	$H_{He}$	$=$	$Id_{1,1}$
(ii)	$K_{She}$	$\sim$	$H_{She}$	$=$	$Id_{1,1}$
(iii)	$K_{Merge-IV}$	$\sim$	$H_{Merge-IV}$	$=$	$F_{Concat}$
(iv)	$K_{Merge-S}$	$\sim$	$H_{Merge-S}$	$=$	$F_{Concat}^{<Id_{2,2}, Id_{1,1}>}$

b. Illustrative Translations

- (i)  $k(\underline{he \ 2 \ smokes}) =$   
 $k(K_{Merge-S}(K_{He}(<2, \emptyset>), <smokes, \emptyset>)) =$   
 $H_{Merge-S}(H_{He}(k(<2, \emptyset>), k(<smokes, \emptyset>))) =$   
 $H_{Merge-S}(H_{He}(x_2), \mathbf{smokes'}) =$   
 $F_{Concat}^{<Id_{2,2}, Id_{1,1}>}(Id_{1,1}(x_2), \mathbf{smokes'}) =$   
 $F_{Concat}^{<Id_{2,2}, Id_{1,1}>}(x_2, \mathbf{smokes'}) =$   
 $(\mathbf{smokes'} x_2)$
- (ii)  $k(\underline{she \ 5 \ loves \ him \ 7}) =$   
 $k(K_{Merge-S}(K_{She}(<5, \emptyset>),$   
 $K_{Merge-IV}(<loves, \emptyset>, K_{He}(<7, \emptyset>)))) =$   
 $H_{Merge-S}(H_{She}(k(<5, \emptyset>)),$   
 $H_{Merge-IV}(k(<loves, \emptyset>), H_{He}(k(<7, \emptyset>)))) =$   
 $H_{Merge-S}(H_{She}(x_5), H_{Merge-IV}(\mathbf{loves'}, H_{He}(x_7))) =$   
 $F_{Concat}^{<Id_{2,2}, Id_{1,1}>}(Id_{1,1}(x_5), F_{Concat}(\mathbf{loves'}, Id_{1,1}(x_7))) =$   
 $F_{Concat}^{<Id_{2,2}, Id_{1,1}>}(x_5, F_{Concat}(\mathbf{loves'}, x_7))) =$   
 $((\mathbf{loves'} x_7) x_5)$

### 3.2 The Syntactic Operation $K_{QI}$ ('Quantifying In')

#### (32) The Rough Idea Behind $K_{QI}$

This will take (i) a quantificational term, and (ii) a sentence containing a pronoun, and output a string where the pronoun is **replaced** with the quantification term.

$$K_{QI}(\text{every president}, \underline{\text{Michelle loves him}_2}) = \underline{\text{Michelle loves every president.}}$$

#### (33) Immediate Issue

To properly define such an operation, there must be some means of specifying *which* pronoun in the sentential argument is to be replaced. *Here, there are two possible paths:*

a. First Implementation ("Universal Grammar"):

$K_{QI}$  is a *ternary* relation, whose first argument is the index of the pronoun to be replaced in the sentential argument.

b. Second Implementation ("PTQ"):

Instead of a single operation  $K_{QI}$ , we have an *infinite* number of operations  $K_{QI,n}$ , where  $n$  is the index of the pronoun to be replaced in the sentential argument.

- In this unit, we'll follow Montague in UG, and pursue the path in (33a).<sup>1</sup>

#### (36) Preliminary Notation: $Pro$ = a meta-variable ranging over *he, she, him, her*

#### (37) The Operation $K_{QI}$

Let  $\alpha, \beta, \gamma$  be trees whose root nodes are ordered pairs. In addition, let  $\alpha', \beta', \gamma'$  be the first members of the root nodes of  $\alpha, \beta, \gamma$  (respectively).

$$K_{QI}(\alpha, \beta, \gamma) = \begin{array}{c} < \delta, QI > \\ \diagdown \quad \diagup \\ \beta \quad \alpha \quad \gamma \end{array}$$

Where  $\delta$  is the result of replacing in  $\gamma'$  the left-most instance of  $Pro \alpha'$  with  $\beta'$ .

---

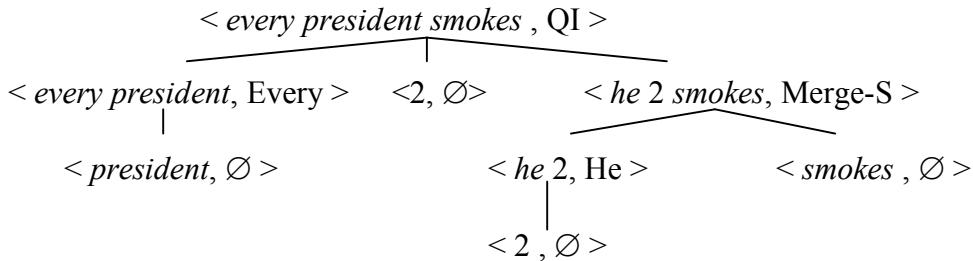
<sup>1</sup> The statement here is somewhat misleading, since Montague didn't introduce a 'quantifying-in' operation in UG. However, the analysis Montague pursues for relative clauses in UG and PTQ faces the same issue in (33). The general approach in (33a) was the one taken up by Montague in UG for the operation forming relative clauses. The approach in (33b) was taken up in PTQ for both 'quantifying-in' and relativization.

(38) **Illustration of  $K_{QI}$**

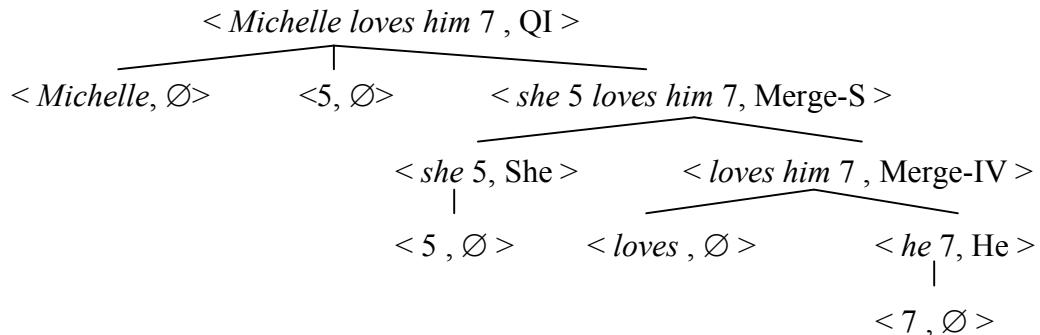
In the examples below, let  $T_1$  be the tree in (29a),  $T_2$  be the tree in (29b). Finally, let  $T_3$  be the tree:  $\langle \text{every president}, \text{Every} \rangle$

$$\begin{array}{c} | \\ \langle \text{president}, \emptyset \rangle \end{array}$$

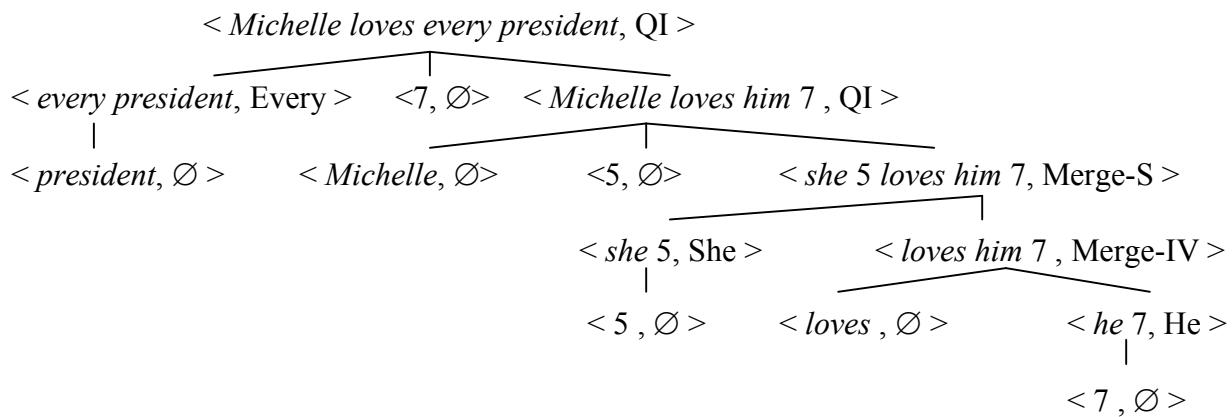
a.  $K_{QI}(\langle 2, \emptyset \rangle, T_3, T_1) =$



b.  $K_{QI}(\langle 5, \emptyset \rangle, \langle \text{Michelle}, \emptyset \rangle, T_2) =$

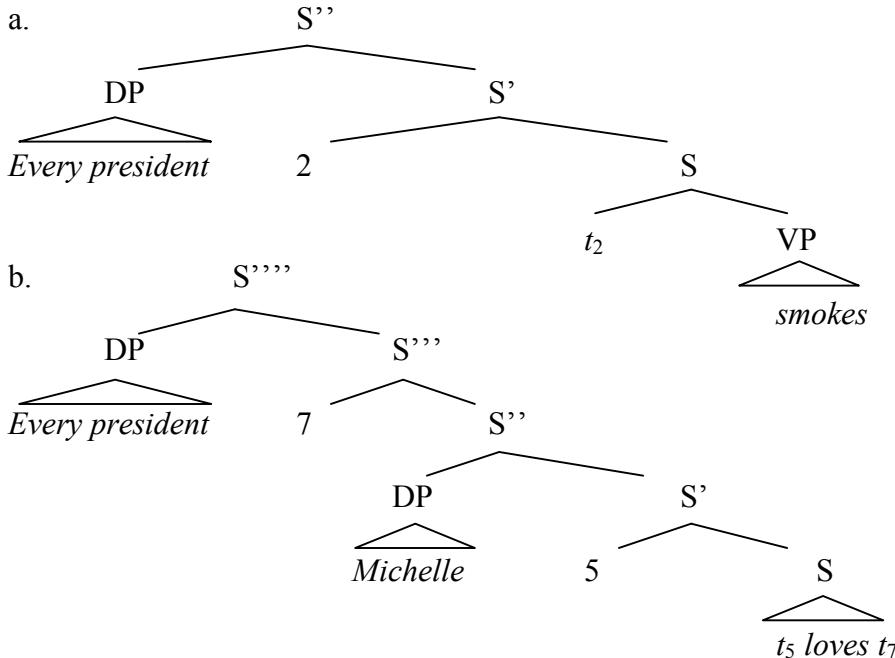


c.  $K_{QI}(\langle 7, \emptyset \rangle, T_3, (38b)) =$



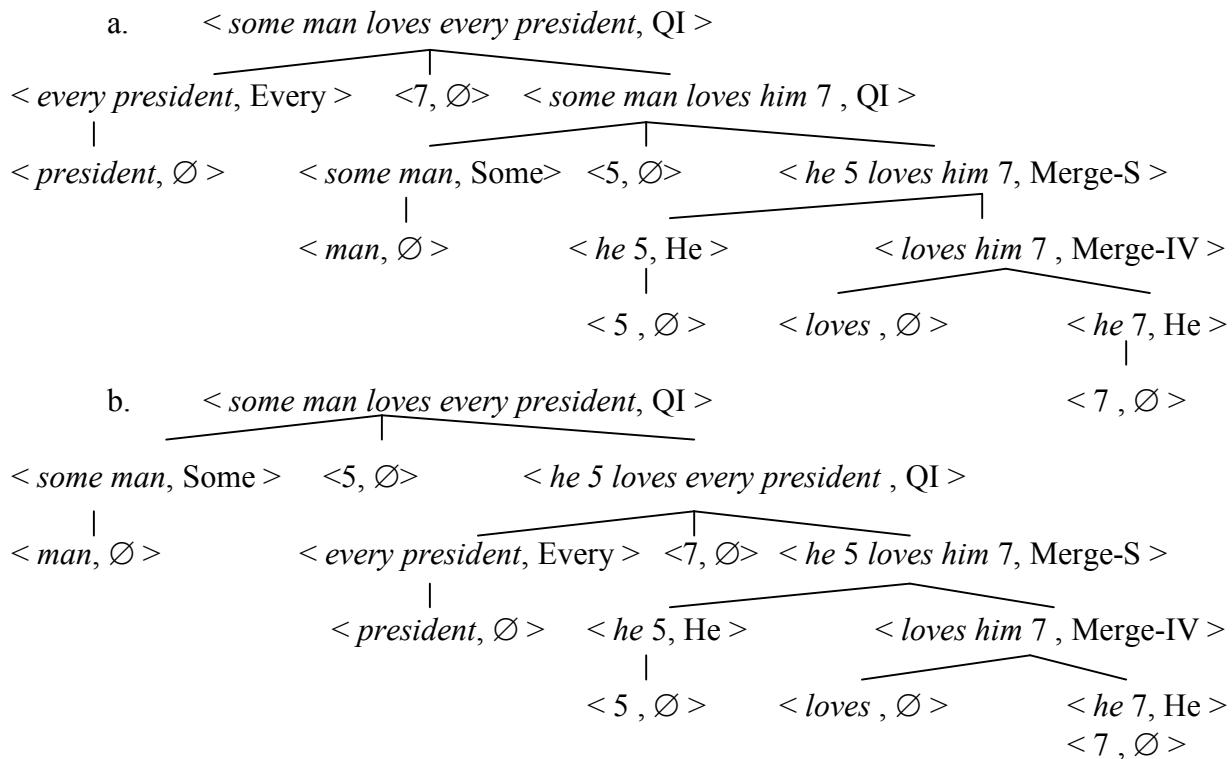
(39) **Remark**

Notice how the analysis tree structures in (38) are geometrically similar to the LFs below, which would be posited in a ‘Heim & Kratzer’-style system! (Hmmmm....)



(40) **The Roots of Quantifier Scope Ambiguities**

Our operation  $K_{QI}$  will be able to output *both* the following analysis tree structures. Note that the relation R in (30) would map both to the string *some man loves every president*.



(41) **Key Observation**

In the system we will ultimately develop from this, the English string *some man loves every president* is syntactically ambiguous.

- One analysis tree that R in (30) maps to this string is derived by QI-ing some man before every president (40a).
- Another analysis tree that R in (30) maps to this string is derived by QI-ing every president before some man (40b).

(42) **Key Idea**

We should design the polynomial operation  $H_{QI}$  so that this syntactic ambiguity will lead to a semantic ambiguity.

$$\begin{aligned} k((40a)) &= \forall x_0 ((\text{president}' x_0) \rightarrow \exists x_1 ((\text{man}' x_1) \& ((\text{loves}' x_0 x_1)))) \\ k((40b)) &= \exists x_0 ((\text{man}' x_0) \& \forall x_1 ((\text{president}' x_1) \rightarrow ((\text{loves}' x_1 x_0)))) \end{aligned}$$

To close out this section, let us introduce the syntactic rule below.

(43) **The Syntactic Rule Employing  $K_{QI}$**

$\langle K_{QI}, \langle IN, T, S \rangle, S \rangle$

'Applying  $K_{QI}$  to an expression of category IN (index), an expression of category T (term), and an expression of category S (sentence), yields an expression of category S.'

Note: Given this rule, the following are members of  $C_S$ : (38a), (38b), (38c), (40a), (40b)

### 3.3 The Polynomial Operation $H_{QI}$ , Corresponding to $K_{QI}$

(44) **What We  $H_{QI}$  to Do For Us**

In general, if  $H_{QI}(x_n, \varphi, \psi) = (\varphi(\lambda x_n \psi))$ , that would work well for our translation.

$$k(\underline{\text{every president smokes}}) = \text{(by (38a))}$$

$$k(K_{QI}(\langle 2, \emptyset \rangle, \underline{\text{every president}}, \underline{\text{he 2 smokes}})) = \text{(by homomorphism property)}$$

$$H_{QI}(k(\langle 2, \emptyset \rangle), k(\underline{\text{every president}}), k(\underline{\text{he 2 smokes}})) = \text{(by (27a), (8), (31b))}$$

<div style="border: 1px solid black; padding: 5px; display: inline-block;">Key Part!</div>	$\left\{ \begin{array}{l} H_{QI}(x_2, (\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))), (\text{smokes}' x_2)) = \text{(by assumption)} \\ ((\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))) (\lambda x_2 (\text{smokes}' x_2))) \Leftrightarrow \text{(by } \lambda\text{-conversion)} \end{array} \right.$
--	---

$$\forall x_0 ((\text{president}' x_0) \rightarrow (\text{smokes}' x_0))$$

(45) **A Polynomial Operation that Would Do the Job**

Let us define  $H_{\text{QI}}$  as the following polynomial operation:  $F_{\text{Concat}} < \text{Id}_{2,3}, F_{\lambda} < \text{Id}_{1,3}, \text{Id}_{3,3} >>$

Illustration:

$$\begin{aligned}
 F_{\text{Concat}} &< \text{Id}_{2,3}, F_{\lambda} < \text{Id}_{1,3}, \text{Id}_{3,3} >> (x_2, (\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))), (\text{smokes}' x_2)) = \\
 F_{\text{Concat}} & ( \text{Id}_{2,3}(x_2, (\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))), (\text{smokes}' x_2)), \\
 F_{\lambda} & ( \text{Id}_{1,3}(x_2, (\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))), (\text{smokes}' x_2)) \\
 & \quad \text{Id}_{3,3}(x_2, (\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))), (\text{smokes}' x_2))) ) = \\
 F_{\text{Concat}} & ( (\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))), F_{\lambda}(x_2, (\text{smokes}' x_2)) ) = \\
 F_{\text{Concat}} & ( (\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))), (\lambda x_2 (\text{smokes}' x_2)) ) = \\
 & ( (\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))) (\lambda x_2 (\text{smokes}' x_2)) )
 \end{aligned}$$

We're almost ready to set up our fragment of English and the translation base. The last ingredient is to find polynomial operations corresponding to  $K_{\text{Some}}$  and  $K_{\text{Every}}$

(46) **The Polynomial Operation  $H_{\text{Some}}$ , Corresponding to  $K_{\text{Some}}$**

a. What We Want  $H_{\text{Some}}$  to Do For Us

In general, if  $H_{\text{Some}}(\varphi) = (\lambda P_0 \exists x_0 ((\varphi x_0) \& (P_0 x_0)))$ , that would work well.

$$k(\underline{\text{some man}}) = k(K_{\text{Some}}(<\text{man}, \emptyset>)) = H_{\text{Some}}(k(<\text{man}, \emptyset>)) = H_{\text{Some}}(\text{man}') = (\lambda P_0 \exists x_0 ((\text{man}' x_0) \& (P_0 x_0)))$$

b. A Polynomial Operation that Would Do the Job

Let us define  $H_{\text{Some}}$  as the following polynomial operation:

$$F_{\lambda} < C_{P0,1}, F_{\exists} < C_{x0,1}, F_{\text{And}} < F_{\text{Concat}} < \text{Id}_{1,1}, C_{x0,1} >, F_{\text{Concat}} < C_{P0,1}, C_{x0,1} >>>$$

Illustration:

$$\begin{aligned}
 F_{\lambda} &< C_{P0,1}, F_{\exists} &< C_{x0,1}, F_{\text{And}} &< F_{\text{Concat}} < \text{Id}_{1,1}, C_{x0,1} >, F_{\text{Concat}} &< C_{P0,1}, C_{x0,1} >>> (\text{man}') = \\
 F_{\lambda} & ( C_{P0,1}(\text{man}'), F_{\exists} ( C_{x0,1}(\text{man}') ), \\
 F_{\text{And}} & ( F_{\text{Concat}} ( \text{Id}_{1,1}(\text{man}'), C_{x0,1}(\text{man}') ), \\
 & \quad F_{\text{Concat}} ( C_{P0,1}(\text{man}'), C_{x0,1}(\text{man}') ) ) ) ) = \\
 F_{\lambda} & ( P_0, F_{\exists} ( x_0, F_{\text{And}} ( F_{\text{Concat}} ( \text{man}', x_0 ), F_{\text{Concat}} ( P_0, x_0 ) ) ) ) ) = \\
 F_{\lambda} & ( P_0, F_{\exists} ( x_0, F_{\text{And}} ( (\text{man}' x_0), (P_0 x_0) ) ) ) ) = \\
 F_{\lambda} & ( P_0, F_{\exists} ( x_0, ((\text{man}' x_0) \& (P_0 x_0)) ) ) ) = \\
 (\lambda P_0 \exists x_0 & ((\text{man}' x_0) \& (P_0 x_0)))
 \end{aligned}$$

(47) **The Polynomial Operation  $H_{\text{Every}}$ , Corresponding to  $K_{\text{Every}}$**

a. What We Want  $H_{\text{Every}}$  to Do For Us

In general, if  $H_{\text{Every}}(\varphi) = (\lambda P_0 \forall x_0 ((\varphi x_0) \rightarrow (P_0 x_0)))$ , that would work well.

$$k(\underline{\text{every man}}) = k(K_{\text{Every}}(<\text{man}, \emptyset>)) = H_{\text{Every}}(k(<\text{man}, \emptyset>)) = H_{\text{Every}}(\text{man}') = (\lambda P_0 \forall x_0 ((\text{man}' x_0) \rightarrow (P_0 x_0)))$$

b. A Polynomial Operation that Would Do the Job

Let us define  $H_{\text{Every}}$  as the following polynomial operation:

$$F_\lambda < C_{P0,1} , F_\forall < C_{x0,1} , F_{\text{If}} < F_{\text{Concat}} < \text{Id}_{1,1} , C_{x0,1} > , F_{\text{Concat}} < C_{P0,1} , C_{x0,1} >>>$$

*Illustration:*

$$\begin{aligned} F_\lambda < C_{P0,1} , F_\forall < C_{x0,1} , F_{\text{If}} < F_{\text{Concat}} < \text{Id}_{1,1} , C_{x0,1} > , F_{\text{Concat}} < C_{P0,1} , C_{x0,1} >>>(\text{man}') &= \\ F_\lambda(C_{P0,1}(\text{man}')) , F_\forall(C_{x0,1}(\text{man}')) , & \\ F_{\text{If}}(F_{\text{Concat}}(\text{Id}_{1,1}(\text{man}'), C_{x0,1}(\text{man}')) , & \\ F_{\text{Concat}}(C_{P0,1}(\text{man}'), C_{x0,1}(\text{man}')))) &= \\ F_\lambda(P_0, F_\forall(x_0, F_{\text{If}}(F_{\text{Concat}}(\text{man}', x_0), F_{\text{Concat}}(P_0, x_0)))) &= \\ F_\lambda(P_0, F_\forall(x_0, F_{\text{If}}(\text{man}' x_0, (P_0 x_0)))) &= \\ F_\lambda(P_0, F_\forall(x_0, ((\text{man}' x_0) \rightarrow (P_0 x_0)))) &= \\ (\lambda P_0 \forall x_0 ((\text{man}' x_0) \rightarrow (P_0 x_0))) & \end{aligned}$$

We're now ready to put all these ideas together into a semantic analysis of English quantification!

4. **The Fragment of English: ‘Mini-English+Qs’ (ME+Q)**

$$(48) \quad \text{Step One: The Category Labels} \quad \Delta \quad = \quad \{\text{TV, IV, S, T, CN, IN, PR}\}$$

(49) **Step Two: The Basic Expressions**

$$\begin{aligned} X_{\text{TV}} &= \{<\text{loves}, \emptyset>\} \\ X_{\text{IV}} &= \{<\text{smokes}, \emptyset>\} \\ X_{\text{T}} &= \{<\text{Barack}, \emptyset>, <\text{Mitt}, \emptyset>, <\text{Michelle}, \emptyset>\} \\ X_{\text{CN}} &= \{<\text{man}, \emptyset>, <\text{president}, \emptyset>\} \\ X_{\text{IN}} &= \{<n, \emptyset> : n \in \mathbb{N}\} \\ X_{\text{PR}} &= \emptyset \\ X_{\text{S}} &= \emptyset \end{aligned}$$

(50) **Step Three: The Syntactic Operations**

a. Operations Unchanged From Disambiguated Mini-English (DME)

- |                      |   |                         |
|----------------------|---|-------------------------|
| $K_{\text{Merge-S}}$ | = | (as defined previously) |
| $K_{\text{Not}}$     | = | (as defined previously) |
| $K_{\text{And}}$     | = | (as defined previously) |
| $K_{\text{If}}$      | = | (as defined previously) |

b. New Syntactic Operations

- |                       |   |                       |
|-----------------------|---|-----------------------|
| $K_{\text{Some}}$     | = | (as defined in (5c))  |
| $K_{\text{Every}}$    | = | (as defined in (5c))  |
| $K_{\text{He}}$       | = | (as defined in (28c)) |
| $K_{\text{She}}$      | = | (as defined in (28c)) |
| $K_{\text{Merge-IV}}$ | = | (as defined in (28c)) |
| $K_{\text{QI}}$       | = | (as defined in (37))  |

$$K_{\text{Or}}(\alpha, \beta) = \langle \alpha' \text{ or } \beta', \text{Or} \rangle$$

$\alpha \swarrow \searrow \beta$

Another new one we'll add,  
just for fun!

(51) **Step Four: The Syntactic Algebra**

$\langle E, K_\gamma \rangle_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If, Or, Some, Every, He, She, QI}\}}$  is the algebra such that:

- a.  $\{ K_\gamma \}_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If, Or, Some, Every, He, She, QI}\}}$  are as defined above, and
- b. E is the smallest set such that
  - (i) For all  $\delta \in \Delta$ ,  $X_\delta \subseteq E$
  - (ii) E is closed under  $\{ K_\gamma \}_{\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If, Or, Some, Every, He, She, QI}\}}$

(52) **Step Five: The Syntactic Rules**

The set  $S_E$  consists of the following triples:

- a.  $\langle K_{\text{Merge-S}}, \langle PR, IV \rangle, S \rangle$
- b.  $\langle K_{\text{Merge-IV}}, \langle TV, PR \rangle, IV \rangle$
- c.  $\langle K_{\text{Not}}, \langle S \rangle, S \rangle$
- d.  $\langle K_{\text{And}}, \langle S, S \rangle, S \rangle$
- e.  $\langle K_{\text{If}}, \langle S, S \rangle, S \rangle$
- f.  $\langle K_{\text{Or}}, \langle S, S \rangle, S \rangle$
- g.  $\langle K_{\text{Some}}, \langle CN \rangle, T \rangle$
- h.  $\langle K_{\text{Every}}, \langle CN \rangle, T \rangle$
- i.  $\langle K_{\text{He}}, \langle IN \rangle, PR \rangle$
- j.  $\langle K_{\text{She}}, \langle IN \rangle, PR \rangle$
- k.  $\langle K_{\text{QI}}, \langle IN, T, S \rangle, S \rangle$

Note: Contrary to the picture in (12)-(14), the rules above do not allow us to  $K_{\text{Merge-S}}$  a term T with an IV to form an S.

- We could easily add such a rule, but will refrain from doing so until later...

(53) **The Disambiguated Language: ‘Disambiguated Mini-English+Qs’ (DME+Q)**

The structure  $\langle E, K_\gamma, X_\delta, S_E, S \rangle_\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If, Or, Some, Every, He, She, QI}\}$ ,  $\delta \in \Delta$  where  $E, K_\gamma, X_\delta, S_E$ , and  $\Delta$  are as defined in (48)-(52).

(54) **Some Illustrative Members of  $C_S$**

Students are encouraged to work out for themselves how the rules in (52) entail that the following are all members of  $C_S$  in DME+Q.

- a.  $\begin{array}{c} < \underset{\diagdown}{\text{he}} \underset{\diagup}{2} \underset{\diagup}{\text{smokes}}, \text{Merge-S} > \\ \diagdown \qquad \diagup \\ < \underset{\mid}{\text{he}} \underset{\mid}{2}, \text{He} > \qquad < \underset{\mid}{\text{smokes}}, \emptyset > \\ | \\ < 2, \emptyset > \end{array}$
- b.  $\begin{array}{c} < \underset{\diagdown}{\text{she}} \underset{\diagup}{5} \underset{\diagup}{\text{loves him}} \underset{\diagup}{7}, \text{Merge-S} > \\ \diagdown \qquad \diagup \qquad \diagup \\ < \underset{\mid}{\text{she}} \underset{\mid}{5}, \text{She} > \qquad < \underset{\mid}{\text{loves him}} \underset{\mid}{7}, \text{Merge-IV} > \\ | \qquad | \\ < 5, \emptyset > \qquad < \underset{\mid}{\text{loves}}, \emptyset > \qquad < \underset{\mid}{\text{he}} \underset{\mid}{7}, \text{He} > \\ | \\ < 7, \emptyset > \end{array}$
- c.  $\begin{array}{c} < \underset{\diagdown}{\text{Michelle}} \underset{\diagup}{\text{loves}} \underset{\diagup}{\text{every}} \underset{\diagup}{\text{president}}, \text{QI} > \\ \diagdown \qquad \diagup \qquad \diagup \\ < \underset{\mid}{\text{every}} \underset{\mid}{\text{president}}, \text{Every} > \qquad < 7, \emptyset > \qquad < \underset{\mid}{\text{Michelle}} \underset{\mid}{\text{loves him}} \underset{\mid}{7}, \text{QI} > \\ | \\ < \underset{\mid}{\text{president}}, \emptyset > \qquad < \underset{\mid}{\text{Michelle}}, \emptyset > \qquad < 5, \emptyset > \qquad < \underset{\mid}{\text{she}} \underset{\mid}{5} \underset{\mid}{\text{loves him}} \underset{\mid}{7}, \text{Merge-S} > \\ | \qquad | \qquad | \qquad | \\ < 5, \emptyset > \qquad < \underset{\mid}{\text{loves}}, \emptyset > \qquad < \underset{\mid}{\text{he}} \underset{\mid}{7}, \text{He} > \\ | \\ < 7, \emptyset > \end{array}$
- d.  $\begin{array}{c} < \underset{\diagdown}{\text{some}} \underset{\diagup}{\text{man}} \underset{\diagup}{\text{loves}} \underset{\diagup}{\text{every}} \underset{\diagup}{\text{president}}, \text{QI} > \\ \diagdown \qquad \diagup \qquad \diagup \\ < \underset{\mid}{\text{every}} \underset{\mid}{\text{president}}, \text{Every} > \qquad < 7, \emptyset > \qquad < \underset{\mid}{\text{some}} \underset{\mid}{\text{man}} \underset{\mid}{\text{loves him}} \underset{\mid}{7}, \text{QI} > \\ | \\ < \underset{\mid}{\text{president}}, \emptyset > \qquad < \underset{\mid}{\text{some}} \underset{\mid}{\text{man}}, \text{Some} > \qquad < 5, \emptyset > \qquad < \underset{\mid}{\text{he}} \underset{\mid}{5} \underset{\mid}{\text{loves him}} \underset{\mid}{7}, \text{Merge-S} > \\ | \qquad | \qquad | \qquad | \\ < \underset{\mid}{\text{man}}, \emptyset > \qquad < \underset{\mid}{\text{he}} \underset{\mid}{5}, \text{He} > \qquad < \underset{\mid}{\text{loves him}} \underset{\mid}{7}, \text{Merge-IV} > \\ | \qquad | \qquad | \\ < 5, \emptyset > \qquad < \underset{\mid}{\text{loves}}, \emptyset > \qquad < \underset{\mid}{\text{he}} \underset{\mid}{7}, \text{He} > \\ | \\ < 7, \emptyset > \end{array}$

(55) **The Fragment of English: Mini-English+Qs (ME+Q)**

Mini-English+Qs is the following language, where R is as defined in (30):

$$\langle\langle E, K_\gamma, X_\delta, S_E, S \rangle\rangle_\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If, Or, Some, Every, He, She, QI}\}, \delta \in \Delta, R \rangle$$

(56) **Some Illustrative Members of C<sub>S</sub> in ME+Q**

- a. *he smokes*
- b. *she loves him*
- c. *Michelle loves every president*
- d. *some man loves every president*

Note: As was observed in (41), sentence (56d) of ME+Q is syntactically ambiguous.

---

**5. The Translation Base from Mini-English+Qs to Politics+λ**

(57) **Step One: The Syntactic Category Mapping**

Let the function  $g: \Delta \rightarrow T \cup \{ \langle \text{var}, \tau \rangle : \tau \in T \}$  be defined as follows:

- a.  $g(\text{TV}) = \langle e, \langle e, t \rangle \rangle$
- b.  $g(\text{IV}) = \langle e, t \rangle$
- c.  $g(S) = t$
- d.  $g(T) = \langle \langle e, t \rangle, t \rangle$
- e.  $g(\text{CN}) = \langle e, t \rangle$
- f.  $g(\text{IN}) = \langle \text{var}, e \rangle$
- g.  $g(\text{PR}) = e$

(58) **Step Two: The Polynomial Operations**

Let  $\{ H_\gamma \}_\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If, Or, Some, Every, He, She, QI}\}$  be the following polynomial operations over the syntactic algebra  $\langle A, F_\gamma \rangle_\gamma \in \{\text{Concat, Not, And, Or, If, } \exists, \forall, \lambda\}$  for Politics+λ.

- a.  $H_{\text{Merge-S}} = F_{\text{Concat}} \langle \text{Id}_{2,2}, \text{Id}_{1,1} \rangle$
- b.  $H_{\text{Merge-IV}} = F_{\text{Concat}}$
- c.  $H_{\text{Not}} = F_{\text{Not}}$
- d.  $H_{\text{And}} = F_{\text{And}}$
- e.  $H_{\text{Or}} = F_{\text{Or}}$
- f.  $H_{\text{If}} = F_{\text{If}}$
- g.  $H_{\text{Some}} = F_\lambda \langle C_{P0,1}, F_\exists \langle C_{x0,1}, F_{\text{And}} \langle F_{\text{Concat}} \langle \text{Id}_{1,1}, C_{x0,1} \rangle, F_{\text{Concat}} \langle C_{P0,1}, C_{x0,1} \rangle \rangle \rangle \rangle \gg \gg$
- h.  $H_{\text{Every}} = F_\lambda \langle C_{P0,1}, F_\forall \langle C_{x0,1}, F_{\text{If}} \langle F_{\text{Concat}} \langle \text{Id}_{1,1}, C_{x0,1} \rangle, F_{\text{Concat}} \langle C_{P0,1}, C_{x0,1} \rangle \rangle \rangle \rangle \gg \gg$
- i.  $H_{\text{He}} = \text{Id}_{1,1}$
- j.  $H_{\text{She}} = \text{Id}_{1,1}$
- k.  $H_{\text{QI}} = F_{\text{Concat}} \langle \text{Id}_{2,3}, F_\lambda \langle \text{Id}_{1,3}, \text{Id}_{3,3} \rangle \rangle$

(59) **Step Three: Checking for Derived Syntactic Rules**

In order to employ the polynomial operations above in our translation base, it must be the case that each of the following are derived syntactic rules of Politics+ $\lambda$ .

Students are encouraged to confirm for themselves whether they are.

- a.  $< F_{\text{Concat}} < \text{Id}_{2,2}, \text{Id}_{1,1} >, < e, < e, t >, t >$  ( $\sim$  Rule (52a))
- b.  $< F_{\text{Concat}}, < < e, < e, t >, e >, < e, t > >$  ( $\sim$  Rule (52b))
- c.  $< F_{\text{Not}}, < t >, t >$  ( $\sim$  Rule (52c))
- d.  $< F_{\text{And}}, < t, t >, t >$  ( $\sim$  Rule (52d))
- e.  $< F_{\text{If}}, < t, t >, t >$  ( $\sim$  Rule (52e))
- f.  $< F_{\text{Or}}, < t, t >, t >$  ( $\sim$  Rule (52f))
- g.  $< F_\lambda < C_{P0,1}, F_\exists < C_{x0,1}, F_{\text{And}} < F_{\text{Concat}} < \text{Id}_{1,1}, C_{x0,1} >, F_{\text{Concat}} < C_{P0,1}, C_{x0,1} >>>, < < e, t >, < < e, t >, t > >$  ( $\sim$  Rule (52g))
- h.  $< F_\lambda < C_{P0,1}, F_\forall < C_{x0,1}, F_{\text{If}} < F_{\text{Concat}} < \text{Id}_{1,1}, C_{x0,1} >, F_{\text{Concat}} < C_{P0,1}, C_{x0,1} >>>, < < e, t >, < < e, t >, t > >$  ( $\sim$  Rule (52h))
- i.  $< \text{Id}_{1,1} < < \text{var}, e > > e >$  ( $\sim$  Rule (52i) and (52j))<sup>2</sup>
- j.  $< F_{\text{Concat}} < \text{Id}_{2,3}, F_\lambda < \text{Id}_{1,3}, \text{Id}_{3,3} >>, < < \text{var}, e >, < < e, t >, t >, t >, t > >$  ( $\sim$  Rule (52k))

(60) **Step Four: The Lexical Translation Function**

The function  $j$  has as its domain  $\{X_\delta\}_{\delta \in \Delta}$ , and consists of the following mappings:

- a.  $j(< \text{loves}, \emptyset >) = \text{loves'}$
- b.  $j(< \text{smokes}, \emptyset >) = \text{smokes'}$
- c.  $j(< \text{Barack}, \emptyset >) = (\lambda P_0 (P_0 \text{ barack}'))$
- d.  $j(< \text{Mitt}, \emptyset >) = (\lambda P_0 (P_0 \text{ mitt}'))$
- e.  $j(< \text{Michelle}, \emptyset >) = (\lambda P_0 (P_0 \text{ michelle}'))$
- f.  $j(< \text{man}, \emptyset >) = \text{man'}$
- g.  $j(< \text{president}, \emptyset >) = \text{president'}$
- h. For all  $n \in \mathbb{N}$ ,  $j(< n, \emptyset >) = x_n (= v_{e,n})$

(61) **Definition of the Translation Base from Mini-English+Qs to Politics+ $\lambda$**

Let  $\mathbf{T}$  be the structure  $< g, H_\gamma, j >_\gamma \in \{\text{Merge-S, Merge-IV, Not, And, If, Or, Some, Every, He, She, QI}\}$ , where  $g$ ,  $H_\gamma$ , and  $j$  are as defined in (57)-(60).  $\mathbf{T}$  is a translation base from Mini-English+Qs to Politics+ $\lambda$ .

We can now use the translation function  $k$  determined by  $\mathbf{T}$  to homomorphically map expressions of DME+Q (analysis trees) to expressions of Politics+ $\lambda$ .

<sup>2</sup> There's actually a problem here for our definition of 'derived syntactic rule'. Although (59i) is true when informally read ('Applying  $\text{Id}_{1,1}$  to an expression of category  $< \text{var}, e >$  yields an expression of category  $e'$ '), it is not strictly speaking a 'derived rule of Politics+ $\lambda$ ', since the label  $< \text{var}, e > \neq e$ . I don't myself know of any possible solutions to this problem. (Montague himself doesn't run into it in UG, for various independent reasons...)

(62) **A Critical Inference Rule When Translating: Alpha-Conversion**

If variable  $v$  is bound in  $\varphi$ , and variable  $v'$  appears nowhere in  $\varphi$ , then  $\varphi$  is logically equivalent to  $[v/v']\varphi$

*Illustration:*  $(\lambda P_0 \exists x_0 ((\text{man}' x_0) \& (P_0 x_0))) \Leftrightarrow (\lambda P_1 \exists x_1 ((\text{man}' x_1) \& (P_1 x_1)))$

(63) **Illustration of the Translation Function  $k$**

Let  $T_1$  be the tree for *some man loves every president* in (40a) [every > some].

- (i)  $k(T_1)$  =
- (ii)  $k(K_{QI}(<7, \emptyset>, K_{\text{Every}}(<\text{president}, \emptyset>), K_{QI}(<5, \emptyset>, K_{\text{Some}}(<\text{man}, \emptyset>), K_{\text{Merge-S}}(K_{\text{He}}(<5, \emptyset>), K_{\text{Merge-IV}}(<\text{loves}, \emptyset>, K_{\text{He}}(<7, \emptyset>)))))))$  =
- (iii)  $H_{QI}(k(<7, \emptyset>), H_{\text{Every}}(k(<\text{president}, \emptyset>)), H_{QI}(k(<5, \emptyset>), H_{\text{Some}}(k(<\text{man}, \emptyset>)), H_{\text{Merge-S}}(H_{\text{He}}(k(<5, \emptyset>)), H_{\text{Merge-IV}}(k(<\text{loves}, \emptyset>), H_{\text{He}}(k(<7, \emptyset>)))))) =$
- (iv)  $H_{QI}(j(<7, \emptyset>), H_{\text{Every}}(j(<\text{president}, \emptyset>)), H_{QI}(j(<5, \emptyset>), H_{\text{Some}}(j(<\text{man}, \emptyset>)), H_{\text{Merge-S}}(H_{\text{He}}(j(<5, \emptyset>)), H_{\text{Merge-IV}}(j(<\text{loves}, \emptyset>), H_{\text{He}}(j(<7, \emptyset>)))))) =$
- (v)  $H_{QI}(x_7, H_{\text{Every}}(\text{president}'), H_{QI}(x_5, H_{\text{Some}}(\text{man}'), H_{\text{Merge-S}}(H_{\text{He}}(x_5), H_{\text{Merge-IV}}(\text{loves}', H_{\text{He}}(x_7)))))) =$
- (vi)  $H_{QI}(x_7, H_{\text{Every}}(\text{president}'), H_{QI}(x_5, H_{\text{Some}}(\text{man}'), ((\text{loves}' x_7) x_5))) =$
- (vii)  $H_{QI}(x_7, (\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))), H_{QI}(x_5, (\lambda P_0 \exists x_0 ((\text{man}' x_0) \& (P_0 x_0))), ((\text{loves}' x_7) x_5))) =$
- (viii)  $H_{QI}(x_7, (\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))), ((\lambda P_0 \exists x_0 ((\text{man}' x_0) \& (P_0 x_0))) (\lambda x_5 ((\text{loves}' x_7) x_5)))) =$
- (ix)  $((\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))) (\lambda x_7 ((\lambda P_0 \exists x_0 ((\text{man}' x_0) \& (P_0 x_0))) (\lambda x_5 ((\text{loves}' x_7) x_5)))))$

(64) **Remark**

The end product of our translation process is the complex lambda-expression in (63ix). However, thanks to  $\alpha$ -conversion and  $\lambda$ -conversion, we can ‘transform’ this into a simpler, logically equivalent formula.

Note:

Because of all the variables shared between the sub-formulae in (63ix), we cannot immediately apply  $\lambda$ -conversion to ‘simplify’ the formula; instead, we must first apply  $\alpha$ -conversion to change the bound variables *so that they aren’t shared between the lambda-expressions.*

(64) Simplifying the Translation in (63)

- (i)  $((\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))) (\lambda x_7 ((\lambda P_0 \exists x_0 ((\text{man}' x_0) \& (P_0 x_0))) (\lambda x_5 ((\text{loves}' x_7) x_5)))) \Leftrightarrow (\alpha\text{-conversion})$
- (ii)  $((\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))) (\lambda x_7 ((\lambda P_1 \exists x_1 ((\text{man}' x_1) \& (P_1 x_1))) (\lambda x_5 ((\text{loves}' x_7) x_5)))) \Leftrightarrow (\lambda\text{-conversion})$
- (iii)  $\forall x_0 ((\text{president}' x_0) \rightarrow ((\lambda x_7 ((\lambda P_1 \exists x_1 ((\text{man}' x_1) \& (P_1 x_1))) (\lambda x_5 ((\text{loves}' x_7) x_5)))) x_0)) \Leftrightarrow (\lambda\text{-conversion})$
- (iv)  $\forall x_0 ((\text{president}' x_0) \rightarrow ((\lambda P_1 \exists x_1 ((\text{man}' x_1) \& (P_1 x_1))) (\lambda x_5 ((\text{loves}' x_0) x_5)))) \Leftrightarrow (\lambda\text{-conversion})$
- (v)  $\forall x_0 ((\text{president}' x_0) \rightarrow \exists x_1 ((\text{man}' x_1) \& ((\lambda x_5 ((\text{loves}' x_0) x_5)) x_1))) \Leftrightarrow (\lambda\text{-conversion})$
- (vi)  $\forall x_0 ((\text{president}' x_0) \rightarrow \exists x_1 ((\text{man}' x_1) \& ((\text{loves}' x_0) x_1)))$

(65) Remark

- Again, strictly speaking, the *translation* of tree (40a) is the complex lambda expression in (63ix).
  - The simpler formula in (64iv) is **not** the translation of (40a).
- (65vi) is however, a logically-equivalent formula, one that allows us to more easily ‘see’ what the semantic value induced by our translation for (40a) is...

(66) Illustration of the Translation Function  $k$

Let  $T_2$  be the tree for *some man loves every president* in (40b) [some > every].

- (i)  $k(T_1) =$
- (ii)  $k( K_{\text{QI}}(<5, \emptyset>, K_{\text{Some}}(<\text{man}, \emptyset>), K_{\text{QI}}(<7, \emptyset>, K_{\text{Every}}(<\text{president}, \emptyset>), K_{\text{Merge-S}}( K_{\text{He}}(<5, \emptyset>), K_{\text{Merge-IV}}(<\text{loves}, \emptyset>, K_{\text{He}}(<7, \emptyset>)))))) =$
- (iii)  $H_{\text{QI}}( k(<5, \emptyset>), H_{\text{Some}}( k(<\text{man}, \emptyset>) ), H_{\text{QI}}( k(<7, \emptyset>), H_{\text{Every}}( k(<\text{president}, \emptyset>) ), H_{\text{Merge-S}}( H_{\text{He}}( k(<5, \emptyset>) ), H_{\text{Merge-IV}}( k(<\text{loves}, \emptyset>), H_{\text{He}}( k(<7, \emptyset>) )) ) ) ) =$

- (iv)  $H_{QI}(j(<5, \emptyset>), H_{\text{Some}}(j(<\text{man}, \emptyset>)),$   
 $H_{QI}(j(<7, \emptyset>), H_{\text{Every}}(j(<\text{president}, \emptyset>)),$   
 $H_{\text{Merge-S}}(H_{\text{He}}(j(<5, \emptyset>)), H_{\text{Merge-IV}}(j(<\text{loves}, \emptyset>), H_{\text{He}}(j(<7, \emptyset>)))))) =$
- (v)  $H_{QI}(x_5, H_{\text{Some}}(\text{man}')),$   
 $H_{QI}(x_7, H_{\text{Every}}(\text{president}')),$   
 $H_{\text{Merge-S}}(H_{\text{He}}(x_5), H_{\text{Merge-IV}}(\text{loves}', H_{\text{He}}(x_7)))) =$
- (vi)  $H_{QI}(x_5, H_{\text{Some}}(\text{man}')),$   
 $H_{QI}(x_7, H_{\text{Every}}(\text{president}')), ((\text{loves}' x_7) x_5)) =$
- (vii)  $H_{QI}(x_5, (\lambda P_0 \exists x_0 ((\text{man}' x_0) \& (P_0 x_0))),$   
 $H_{QI}(x_7, (\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))), ((\text{loves}' x_7) x_5)) =$
- (viii)  $H_{QI}(x_5, (\lambda P_0 \exists x_0 ((\text{man}' x_0) \& (P_0 x_0))),$   
 $((\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))) (\lambda x_7 ((\text{loves}' x_7) x_5)))) =$
- (ix)  $((\lambda P_0 \exists x_0 ((\text{man}' x_0) \& (P_0 x_0)))$   
 $(\lambda x_5 ((\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))) (\lambda x_7 ((\text{loves}' x_7) x_5)))) =$

Once again, we can apply  $\alpha$ -conversion and  $\lambda$ -conversion to ‘transform’ the translation in (ix) into a simpler, logically-equivalent formula

- (x)  $((\lambda P_0 \exists x_0 ((\text{man}' x_0) \& (P_0 x_0)))$   
 $(\lambda x_5 ((\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))) (\lambda x_7 ((\text{loves}' x_7) x_5)))) \Leftrightarrow (\alpha\text{-conversion})$
- (xi)  $((\lambda P_0 \exists x_0 ((\text{man}' x_0) \& (P_0 x_0)))$   
 $(\lambda x_5 ((\lambda P_1 \forall x_1 ((\text{president}' x_1) \rightarrow (P_1 x_1))) (\lambda x_7 ((\text{loves}' x_7) x_5)))) \Leftrightarrow (\lambda\text{-conversion})$
- (xii)  $\exists x_0 ((\text{man}' x_0) \&$   
 $((\lambda x_5 ((\lambda P_1 \forall x_1 ((\text{president}' x_1) \rightarrow (P_1 x_1))) (\lambda x_7 ((\text{loves}' x_7) x_5))) x_0)) \Leftrightarrow (\lambda\text{-conversion})$
- (xiii)  $\exists x_0 ((\text{man}' x_0) \&$   
 $((\lambda P_1 \forall x_1 ((\text{president}' x_1) \rightarrow (P_1 x_1))) (\lambda x_7 ((\text{loves}' x_7) x_0))) \Leftrightarrow (\lambda\text{-conversion})$
- (xiv)  $\exists x_0 ((\text{man}' x_0) \&$   
 $\forall x_1 ((\text{president}' x_1) \rightarrow ((\lambda x_7 ((\text{loves}' x_7) x_0)) x_1))) \Leftrightarrow (\lambda\text{-conversion})$
- (xv)  $\exists x_0 ((\text{man}' x_0) \& \forall x_1 ((\text{president}' x_1) \rightarrow ((\text{loves}' x_1) x_0)))$

Again, the formula in (63xiv) is not the *translation* of tree (40b); the translation is (63ix).  
*However (63xv) is a simpler, logically-equivalent expression...  
Thus, it offers a more ‘transparent’ representation of the meaning assigned to (40b)*

## 6. Inducing an Interpretation for Mini-English+Q

If we compose together our translation function  $k$  and our meaning assignment  $h$  for Politics+ $\lambda$ , we now (thanks to our general theory of translation), obtain a meaning assignment for DME+Q.

### (67) Illustration of the Induced Interpretation of Mini-English+Q

Let  $T_1$  be the tree for *some man loves every president* in (40a) [every > some].

- (i)  $h \circ k(T_1)$  = (by definition of composition)
- (ii)  $h(k(T_1))$  = (by (63))
- (iii)  $h(((\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))) (\lambda x_7 ((\lambda P_0 \exists x_0 ((\text{man}' x_0) \& (P_0 x_0))) (\lambda x_5 ((\text{loves}' x_7) x_5)))))) =$  (by (64))
- (iv)  $h(\forall x_0 ((\text{president}' x_0) \rightarrow \exists x_1 ((\text{man}' x_1) \& ((\text{loves}' x_0) x_1))))$   
= (by definition of interpretation **B** in (53) of previous handout)
- (v) The function  $A$  such that if  $g \in J$ ,  $A(g) = 1$  iff for all  $x \in D_{e,E}$ ,  
if  $k(x) = 1$  then there exists a  $y \in D_{e,E}$  such that  $i(y) = 1$  and  $j(x)(y)$   
= (by definition of interpretation **B** in (53) of previous handout)
- (vi) The function  $A$  such that if  $g \in J$ ,  $A(g) = 1$

Note:

In our induced interpretation of Mini-English, tree (40a) – the ‘inverse-scope reading of *some man loves every president*’ – is interpreted as being true (relative to any variable assignment).

### (68) Illustration of the Induced Interpretation of Mini-English+Q

Let  $T_2$  be the tree for *some man loves every president* in (40b) [some > every].

- (i)  $h \circ k(T_2)$  = (by definition of composition)
- (ii)  $h(k(T_2))$  = (by (66))
- (iii)  $h(((\lambda P_0 \exists x_0 ((\text{man}' x_0) \& (P_0 x_0))) (\lambda x_5 ((\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))) (\lambda x_7 ((\text{loves}' x_7) x_5)))))) =$  (by (66))
- (iv)  $h(\exists x_0 ((\text{man}' x_0) \& \forall x_1 ((\text{president}' x_1) \rightarrow ((\text{loves}' x_1) x_0)))) =$  (by def. of **B**)
- (v) The function  $E$  such that if  $g \in J$ ,  $E(g) = 1$  iff there is an  $x \in D_{e,E}$  such that  
 $i(x) = 1$  and for all  $y \in D_{e,E}$ , if  $k(y) = 1$  then  $j(y)(x) = 1$  = (by def. of **B**)
- (vi) The function  $E$  such that if  $g \in J$ ,  $E(g) = 1$

Note:

In our induced interpretation of Mini-English, tree (40b) – the ‘surface-scope reading of *some man loves every president*’ – is interpreted as being true (relative to any variable assignment).

## 7. Summary

In these notes, we’ve accomplished the following:

- Developed a fragment of English (ME+Q) which contains quantificational terms (quantificational NPs)
- Provided a translation base homomorphically mapping the expressions of DME+Q to expressions of Politics+ $\lambda$ .
- Via our interpretation for Politics+ $\lambda$  defined in the last handout, obtained an ‘induced’ interpretation for ME+Q.

The resulting system has the following advantageous properties:

- Interprets sentences where quantificational terms are in direct object position
- Correctly / automatically predicts quantifier scope ambiguities in sentences containing more than one quantifier.
- Correctly / automatically predicts quantificational binding of pronouns (HOMEWORK!)

## What’s Next on the Agenda?

- Thus far, all our syntactic and semantic analyses have been within framework as presented in Montague’s paper “Universal Grammar.”
- In his paper “PTQ”, however, Montague employs a relatively simplified presentation of the system, one that allows for a much more transparent / readable treatment of *opaque / intensional contexts*.
- Therefore, to build towards that, I will next ‘transform’ the system we developed in the last few handouts into a system akin to that presented in PTQ.
  - Again, the substance of the analysis will remain the same (as you’ll see)
  - All that really differs is the notation / presentation employed...

### Problem Set on the Analysis of Quantification

The exercises below make reference to the handout “An Algebraic Approach to Quantification and Lambda Abstraction: Applications to the Analysis of English.”

#### (1) Pronominal Binding in Our Fragment of English

- a. Please show how our English fragment in (55) and our translation base in (61) together predict that (i) receives a translation logically equivalent to (ii).
  - (i) *Michelle loves some man and he smokes.*
  - (ii)  $\exists x_0 ((\text{man}' x_0) \& (((\text{loves}' x_0) \text{ michelle}') \& (\text{smokes}' x_0)))$
- b. Does the sentence below raise any problems for our system?
  - (i) *Michelle loves every man and he smokes.*

#### (2) Another Exercise in Indirect and Direct Interpretation

- a. Minimally alter our English fragment in (55) so that its expressions now include strings like *no man smokes* and *Michelle loves no man*.
- b. Minimally alter our translation base in (61) so that strings like *no man smokes* receive appropriate translations in Politics+ $\lambda$ .

**Note:** For this exercise, you don't have to show that anything is a derived syntactic rule.

- c. Please show how the new translation base, along with the interpretation for Politics+ $\lambda$  presented in (53) of the handout “Fregean Interpretations” assign a meaning to the sentence *no man smokes*.
- d. Given your proposed translation base, construct a direct interpretation of Mini-English+Qs.

### (3) One Final Exercise on Direct Interpretation

- a. Minimally alter our English fragment in (55) so that its expressions now include strings like *most man smokes* and *Michelle loves most man*.

**Note:** For this exercise, please disregard the need for *most* in English to appear with a plural NP.

- b. Provide a *direct* interpretation of the resulting fragment.

**HINT:**

For this problem, you should aim to predict that *most man smokes* has a meaning that maps a variable assignment  $g$  to 1 iff the set of men who smoke is larger than the set of men who don't smoke.

That is, it maps  $g$  to 1 iff: 
$$\frac{|\{x : x \text{ in } D_{e,E} \& h(\text{man})(g)(x) = 1 \& h(\text{smoke})(g)(x) = 1\}| > |\{x : x \text{ in } D_{e,E} \& h(\text{man})(g)(x) = 1 \& h(\text{smoke})(g)(x) = 0\}|}{}$$

**Fun Fact:**

- There is no formula of Politics+ $\lambda$  that will have the meaning outlined above.
- Consequently, we cannot use Politics+ $\lambda$  to provide an *indirect* interpretation of the fragment in (3a).
- This highlights the way in which our ability to do *indirect* interpretation of natural language is limited by the logical languages we have at our disposal, **while there are no such limits on our ability to do *direct* interpretation...**

**Unit 6:**  
**The Proper Treatment of Quantification in Ordinary**  
**English**

## First Steps Towards PTQ: A New Presentation of Our System for Quantifiers

### (1) From UG to PTQ

- By far and away, the most cited paper by Montague is “The Proper Treatment of Quantification in Ordinary English” (PTQ).
- When compared to “Universal Grammar”, PTQ is a relatively *informal* presentation of Montague’s key ideas.
  - It was written as a paper for the 1970 Stanford Workshop on Grammar and Semantics.
- Consequently, PTQ doesn’t hold to letter of the UG system and its notations, and introduces certain simplifications...
  - Many aspects of Montague’s analysis in PTQ can be easily rephrased in the strictly algebraic UG framework
  - **Others, as we will see, cannot (without sacrificing some elegance)**

*In these notes, we will take the analysis of English quantification developed in the last few handouts, and ‘convert’ it into a format closer to what is found in PTQ...*

- Many of the changes are simply superficial ones of terminology and notation...

### 1. The Fragment ‘Mini-English+Q’: PTQ-Style Presentation

#### (2) Redefinition: Category

In PTQ, the indices that we’ve been referring to as ‘category labels’ ( $\Delta$  in UG) are instead dubbed simply ‘categories’. The set of ‘categories’ is represented as *Cat* ( $\sim\Delta$  in UG)

- This more closely approximates the terminology in generative linguistics.

#### (3) The Categories of ME+Q: $Cat = \{TV, IV, S, T, CN, PR\}$

#### (4) New Notation: Basic Expressions

In PTQ,  $B_A$  is the set of basic expressions of category A. ( $\sim X_\delta$  in UG)

#### (5) The Basic Expressions of ME+Q

$B_{TV}$	=	{ loves }
$B_{IV}$	=	{ smokes }
$B_T$	=	{ Barack, Mitt, Michelle }
$B_{CN}$	=	{ man, president, woman }
$B_{PR}$	=	{ he $n : n \in \mathbb{N}$ } $\cup$ { she $n : n \in \mathbb{N}$ }
$B_S$	=	$\emptyset$

(6) **Remarks**

- Our English expressions are now boldfaced rather than italicized.
- Following PTQ (rather than UG), pronouns are now primitive expressions, rather than ones syntactically derived from indices.
- To set up something interesting for later, we've also added **woman** to our set of CNs.
- **Note that our English expressions are now *strings* again, rather than trees. As we'll see, this won't pose any problems for the PTQ 'version' of MG....**

(7) **New Terminology: Phrases**

In PTQ, the sets that we've been referring to as (e.g.) 'the category  $\delta$ ' ( $C_\delta$  in UG) are instead dubbed 'the phrases of category  $\delta$ '.

- $P_A =$  The phrases of category A

(8) **The Syntactic Rules**

As in UG, the set  $\bigcup_{A \in \text{Cat}} P_A$  ( $\sim \bigcup_{\delta \in \Delta} C_\delta$  in UG) will be defined via the *syntactic rules*.

- In PTQ, however, a 'syntactic rule' is a rather different object from that in UG.
- As shown below, in PTQ, the 'rules' incorporate *both* (i) the definition of the syntactic operations in the language, and (ii) the information contained in a UG 'rule'.

(9) **The Rule S1**

- a. The Rule:  $B_A \subseteq P_A$ , for every category A.

b. Remarks:

- This is clearly not a 'syntactic rule', in the sense found in UG.
- What S1 does is move to the system of 'syntactic rules' a condition that was (in UG) part of the general definition of the set of categories ( $\bigcup_{\delta \in \Delta} C_\delta$ )

(10) **The Rule S2**

a. The Rule:

If  $\zeta \in P_{\text{CN}}$ , then  $F_0(\zeta), F_2(\zeta) \in P_T$ , where  $F_0(\zeta) = \text{every } \zeta$  and  $F_2(\zeta) = \text{some } \zeta$

b. Remarks:

- Again, this is not a 'syntactic rule', in the sense found in UG.
- What S2 does is combine together the information we had earlier factored out into (i) the definitions of  $K_{\text{Every}}$  and  $K_{\text{Some}}$ , and (ii) the rules  $\langle K_{\text{Every}}, \langle \text{CN} \rangle, T \rangle$  and  $\langle K_{\text{Some}}, \langle \text{CN} \rangle, T \rangle$

Note:

- Following Montague in PTQ (and UG), the indices on our rules will no longer be evocative mnemonics. Rather, they will simply be numerals.
- I will also be numbering both **rules** and **operations** in a way that matches the corresponding rules and operations in PTQ (thus,  $F_0$  for **every** and  $F_2$  for **some**).

(11) **The Rule S4**

- a. The Rule: If  $\alpha \in P_{PR}$  and  $\delta \in P_{IV}$ , then  $F_4(\alpha, \delta) \in P_S$ , where  $F_4(\alpha, \delta) = \alpha \ \delta$

- b. Remarks:

Again, as with S2 and the rules below, this ‘syntactic rule’ combines together the information that in the UG system was factored out into:

- The definition of  $K_{Merge-S}$ , and
- The rule  $\langle K_{Merge-S}, \langle PR, IV \rangle, S \rangle$

(12) **The Rule S5**

- a. The Rule:

If  $\delta \in P_{TV}$  and  $\beta \in P_{PR}$ , then  $F_5(\delta, \beta) \in P_{IV}$ , where  $F_5(\delta, \beta) = \delta \ \text{him } n$  if  $\beta$  has the form **he n**,  $F_5(\delta, \beta) = \delta \ \text{her } n$  if  $\beta$  has the form **she n**, and  $F_5(\delta, \beta) = \delta \ \beta$  otherwise.

- b. Remarks: Again, this rule combines together the following information:

- The definition of  $K_{Merge-IV}$
- The rule  $\langle K_{Merge-IV}, \langle TV, PR \rangle, IV \rangle$

(13) **The Rule S11**

- a. The Rule:

If  $\varphi, \psi \in P_S$ , then  $F_8(\varphi, \psi), F_9(\varphi, \psi) \in P_S$ , where  $F_8(\varphi, \psi) = \varphi \ \text{and } \psi$ , and  $F_9(\varphi, \psi) = \varphi \ \text{or } \psi$ .

- b. Remarks:

- Again, we see how this one ‘syntactic rule’ contains both (i) the definition of the syntactic operations  $F_8$  ( $\sim K_{And}$ ) and  $F_9$  ( $\sim K_{Or}$ ), and (ii) the UG-syntactic rules references those operations.
- I will follow Montague in PTQ by only having these rules and operations; I will henceforth drop the operations  $K_{If}, K_{Not}$ , and the rules involving them.
  - **After all, they give us horrible analyses of conditionals and negation in English ; )**

(14) **The Rule S14**

a. The Rule:

If  $\alpha \in P_T$  and  $\varphi \in P_S$ , then  $F_{10,n}(\alpha, \varphi) \in P_S$ , where  $F_{10,n}(\alpha, \varphi)$  comes from  $\varphi$  by replacing the first occurrence of **he n**, **him n**, **she n**, or **her n**, by  $\alpha$ .

b. Remarks:

- Again, this one ‘syntactic rule’ combines together both:
  - (i) The definition of the operations  $F_{10,n}(\alpha, \varphi)$  ( $\sim K_{QI}(<n, \emptyset>, \alpha, \varphi)$ )
  - (ii) The UG-style rule  $< F_{10,n}, < T, S >, S >$
- Note that this rule doesn’t appeal to a *single* syntactic operation, but a whole infinite family of them,  $\{ F_{10,n} : n \in \mathbb{N} \}$

*With the rules above, we are able to offer the following definition of the set of phrases for ME+Q*

(15) **The Phrases of Mini-English+Q**

$\{ P_A \}_{A \in Cat}$  is the smallest family of sets indexed by  $Cat$  such that S1-S14 are true.

(16) **The Meaningful Expressions of English**

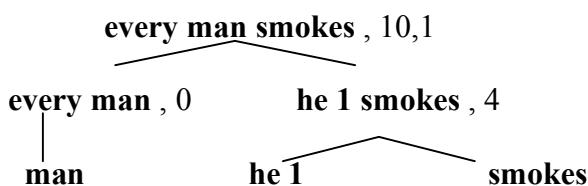
$\varphi$  is a meaningful expression of ME+Q if there is an  $A \in Cat$  such that  $\varphi \in P_A$ .

(17) **Some Illustrative Members of  $P_S$**

a. **Every man smokes**

- (i) **man**  $\in B_{CN}$ , **smokes**  $\in B_{IV}$ , **he 1**  $\in B_{PR}$  (by (5))
- (ii) **man**  $\in P_{CN}$ , **smokes**  $\in P_{IV}$ , **he 1**  $\in P_{PR}$  (by S1)
- (iii) **he 1 smokes**  $\in P_S$  (by S4)
- (iv) **every man**  $\in P_T$  (by S2)
- (v) **every man smokes**  $\in P_S$  (by S14)

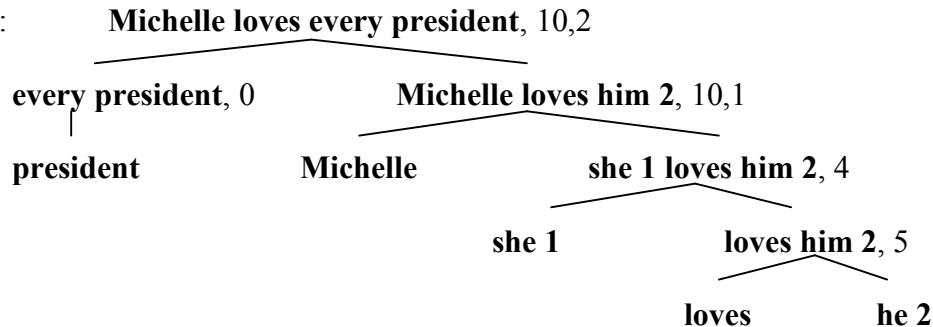
Analysis Tree:



b. Michelle loves every president

- (i) **Michelle**  $\in B_T$ , **president**  $\in B_{CN}$ , **loves**  $\in B_{TV}$ , **she 1, he 2**  $\in B_{PR}$  (5)
- (ii) **Michelle**  $\in P_T$ , **president**  $\in P_{CN}$ , **loves**  $\in P_{TV}$ , **she 1, he 2**  $\in P_{PR}$  (S1)
- (iii) **loves him 2**  $\in P_{IV}$  (S5)
- (iv) **she 1 loves him 2**  $\in P_S$  (S4)
- (v) **Michelle loves him 2**  $\in P_S$  (S14)
- (vi) **Michelle loves every president**  $\in P_S$  (S14)

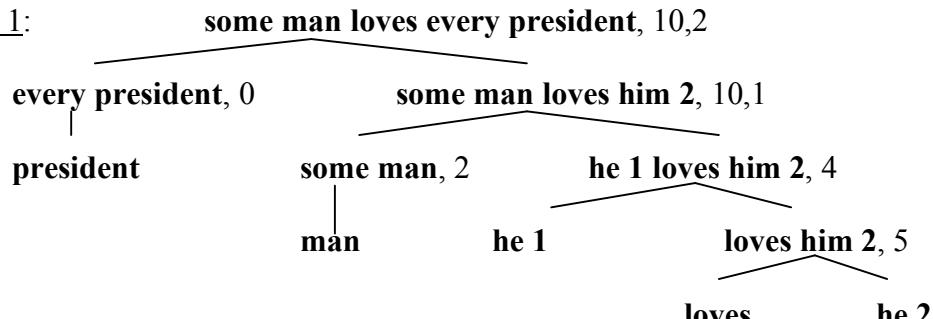
Analysis Tree:



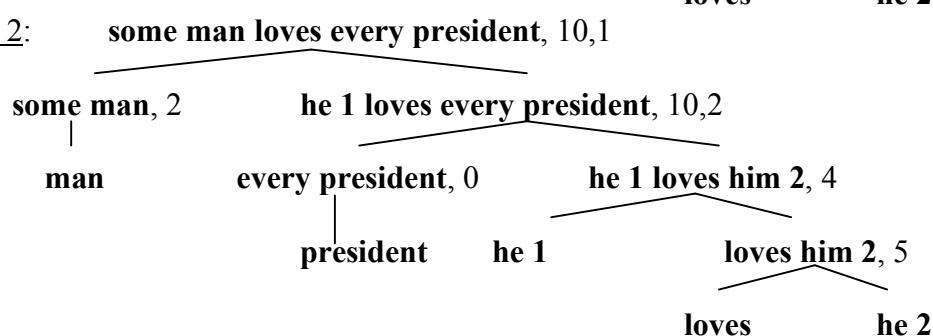
c. Some man loves every president.

- (i) **man, president**  $\in B_{CN}$ , **loves**  $\in B_{TV}$ , **he 1, he 2**  $\in B_{PR}$  (5)
- (ii) **man, president**  $\in P_{CN}$ , **loves**  $\in P_{TV}$ , **he 1, he 2**  $\in P_{PR}$  (S1)
- (iii) **loves him 2**  $\in P_{IV}$  (S5)
- (iv) **he 1 loves him 2**  $\in P_S$  (S4)
- (v) **some man, every president**  $\in P_T$  (S2)
- (vi) **some man loves him 2**  $\in P_S$  / **he 1 loves every president**  $\in P_S$  (S14)
- (vii) **some man loves every president**  $\in P_S$  (S14)

Analysis Tree 1:



Analysis Tree 2:



(18) **Key Observation**

As shown in (17c), there are meaningful expressions of ME+Q that are genuinely syntactically ambiguous.

- ME+Q as defined above would not be a ‘disambiguated language’ in the sense of UG.

(19) **Quote by Montague**

“Thus our fragment admits genuinely (that is, semantically) ambiguous sentences. If it were desired to construct a corresponding unambiguous language, it would be convenient to take the analysis trees themselves as the expressions of that language; it would then be obvious how to characterize...the structural operations of that language and the correspondence relation between its expressions and those of ordinary English. For present purposes, however, no such construction is necessary.” (Montague 1974; p. 23).

No such construction is necessary?...

As we will see shortly, translation in PTQ is simply a *relation*, and need not be a function

---

## 2. The Logical Language TL: PTQ-Style Presentation

(20) **Logical Languages in PTQ: First Key Difference**

- In PTQ, Montague does not represent the logical translation language as a ‘disambiguated language’ in the sense of UG.
  - Rather, he gives a (relatively) simple, recursive definition of its syntax.
- In PTQ, Montague does not represent the semantics of the logical translation language as a ‘(Fregean) interpretation’ in the sense of UG.
  - Rather, he gives a (relatively) simple, model-theoretic semantics.
- **However, given the equivalences between these systems for syntax and semantics, it is possible (though laborious) to convert between them (see UG)**
  - **Thus, there is no substantive difference in the theory of the syntax/semantics of the logical translation language (the presentation in PTQ is just ‘simpler’)**

(21) **Logical Languages in PTQ: The Second Key Difference**

- Up to now, we’ve been translating our natural language into a ‘tailor-made’ logical language with a finite set of constants.
- **Technically speaking, in PTQ (and UG), Montague doesn’t do that, but rather lays out a logical language with an infinite set of constants**
  - That is, in PTQ and UG, **man**’ isn’t the *translation* of **man**, but rather a meta-language abbreviation for “whatever constant *c* is the translation of **man**”

In what follows, we define the syntax and semantics of a single language, dubbed ‘**Typed Logic**’.

(22) **The Vocabulary of Typed Logic (TL)**

- a. The Logical Constants:
  - (i) *Sentence Connectives:*  $\sim, \&, \vee, \rightarrow$
  - (ii) *Quantifiers:*  $\forall, \exists$
  - (iii) *Lambda Operator:*  $\lambda$
- b. The Syntactic Symbols:  $(, )$
- c. The Non-Logical Constants:
  - (i) *Constants:*  
For every type  $\tau \in T$ , a **countably infinite** set of constants of type  $\tau$ :  

$$CON_{\tau} = \{ c_{\tau, n} : n \in \mathbb{N} \}$$
  - (ii) *Variables:*  
For every type  $\tau \in T$ , a **countably infinite** set of variables of type  $\tau$ :  

$$VAR_{\tau} = \{ v_{\tau, n} : n \in \mathbb{N} \}$$

(23) **Meta-Language Abbreviations for Variables and Constants**

Although our variables ‘officially’ all look like those in (22c), to save space we will make use of the following meta-language abbreviations:

- |                    |                                       |  |
|--------------------|---------------------------------------|--|
| a. $x_n = v_{e,n}$ | b. $P_n = v_{\langle e,t \rangle, n}$ |  |
| c. $a_n = c_{e,n}$ | d. $Q_n = c_{\langle e,t \rangle, n}$ | e. $R_n = c_{\langle e, \langle e,t \rangle \rangle, n}$ |

(23) **The Syntax of TL**

- a. If  $\varphi \in CON_{\tau}$  or  $\varphi \in VAR_{\tau}$ , then  $\varphi \in ME_{\tau}$
- b. If  $\varphi \in ME_{\sigma, \tau}$  and  $\psi \in ME_{\sigma}$ , then  $(\varphi \psi) \in ME_{\tau}$
- c. If  $\varphi, \psi \in ME_t$ , then
  - (i)  $\sim\varphi \in ME_t$
  - (ii)  $(\varphi \& \psi) \in ME_t$
  - (iii)  $(\varphi \vee \psi) \in ME_t$
  - (iv)  $(\varphi \rightarrow \psi) \in ME_t$
- d. If  $v \in VAR_{\tau}$ , and  $\varphi \in ME_t$ , then
  - (i)  $\exists v \varphi \in ME_t$
  - (ii)  $\forall v \varphi \in ME_t$
- e. If  $v \in VAR_{\sigma}$ , and  $\varphi \in ME_{\tau}$ , then  $(\lambda v \varphi) \in ME_{\langle \sigma, \tau \rangle}$

(24) Some Illustrative Meaningful Expressions of TL

- a.  $\sim ((R_3 a_2) a_1)$   $[\sim ((\text{loves}' \text{ mitt}') \text{ barack}') \text{ in Politics} + \lambda]$
- b.  $\forall x_3 ((Q_2 x_3) \rightarrow \sim ((R_3 a_2) x_3))$   $[\forall x_3 ((\text{smokes}' x_3) \rightarrow \sim ((\text{loves}' \text{ mitt}') x_3))]$
- c.  $\exists P_4 (P_4 a_3)$   $[\exists P_4 (P_4 \text{ michelle}') \text{ in Politics} + \lambda]$
- d.  $((\lambda x_3 (Q_3 x_3)) a_2)$   $[((\lambda x_3 (\text{man}' x_3)) \text{ mitt}') \text{ in Politics} + \lambda]$
- e.  $(\lambda P_4 \forall x_3 ((Q_3 x_3) \rightarrow (P_4 x_3)))$   $[(\lambda P_4 \forall x_3 ((\text{man}' x_3) \rightarrow (P_4 x_3))), \text{ Politics} + \lambda]$

*Our semantics for TL is going to exactly follow our model-theoretic semantics for Politics+λ*

(25) The Semantics of TL, Part 1: The Denotations Based on a Set E

Let T be the set of types and E be some non-empty set (of entities). If  $\tau \in T$ , then the set  $D_{\tau, E}$  of *denotations of type τ based on E* is defined as follows:

- (i)  $D_{e, E} = E$
- (ii)  $D_{t, E} = \{0, 1\}$
- (iii) If  $\sigma, \tau \in T$ , then  $D_{\langle \sigma, \tau \rangle, E} =$  the set of functions from  $D_{\sigma, E}$  to  $D_{\tau, E}$

(26) The Semantics of TL, Part 2: A Model for TL

A model  $\mathcal{M}$  for TL is a pair  $\langle E, I \rangle$  consisting of:

- a. A *non-empty* set E, called the ‘domain of  $\mathcal{M}$ ’
- b. A function I, whose domain is equal to (i) and whose range satisfies the condition in (ii).
  - (i) *Domain of I:*  $\bigcup_{\tau \in T} \text{CON}_{\tau}$
  - (ii) *Condition on Range of I:* If  $\alpha \in \text{CON}_{\tau}$ , then  $I(\alpha) \in D_{\tau, E}$

(27) The Semantics of TL, Part 3: Variable Assignments

Let  $\mathcal{M}$  be a model  $\langle E, I \rangle$  of TL. Then g is a *variable assignment (based on  $\mathcal{M}$ )* if its domain is equal to (i) and its range satisfies the property in (ii).

- (i) *Domain of g:*  $\bigcup_{\tau \in T} \text{VAR}_{\tau}$
- (iii) *Condition on Range of g:* If  $\alpha \in \text{VAR}_{\tau}$ , then  $g(\alpha) \in D_{\tau, E}$

(28) **The Semantics of TL, Part 4: Interpretation w.r.t. Model and Variable Assignment**

Let  $\mathcal{M}$  be a model  $\langle E, I \rangle$  for TL and  $g$  be a variable assignment based on  $\mathcal{M}$ . The interpretation (a.k.a. denotation) of a meaningful expression of TL relative to  $\mathcal{M}$  and  $g$   $[[\cdot]]^{M,g}$  is defined as follows:

- a. If  $v \in \bigcup_{\tau \in T} VAR_{\tau}$ , then  $[[v]]^{M,g} = g(v)$
- b. If  $\alpha \in \bigcup_{\tau \in T} CON_{\tau}$ , then  $[[\alpha]]^{M,g} = I(\alpha)$
- c. If  $\varphi = (\psi \chi)$ , then  $[[\varphi]]^{M,g} = [[\psi]]^{M,g} [[\chi]]^{M,g}$
- d. If  $\varphi = \sim\psi$ , then  $[[\varphi]]^{M,g} = 1$  iff  $[[\psi]]^{M,g} = 0$
- e. If  $\varphi = (\psi \& \chi)$ , then  $[[\varphi]]^{M,g} = 1$  iff  $[[\psi]]^{M,g} = 1$  and  $[[\chi]]^{M,g} = 1$
- f. If  $\varphi = (\psi \vee \chi)$ , then  $[[\varphi]]^{M,g} = 1$  iff  $[[\psi]]^{M,g} = 1$  or  $[[\chi]]^{M,g} = 1$
- g. If  $\varphi = (\psi \rightarrow \chi)$ , then  $[[\varphi]]^{M,g} = 1$  iff  $[[\psi]]^{M,g} = 0$  or  $[[\chi]]^{M,g} = 1$
- h. If  $\varphi = \exists v\psi$  and  $v \in VAR_{\tau}$ , then  $[[\varphi]]^{M,g} = 1$  iff  
there is an  $a \in D_{\tau,E}$  such that  $[[\psi]]^{M,g(v/a)} = 1$
- i. If  $\varphi = \forall v\psi$  and  $v \in VAR_{\tau}$ , then  $[[\varphi]]^{M,g} = 1$  iff for all  $a \in D_{\tau,E}$ ,  $[[\psi]]^{M,g(v/a)} = 1$
- j. If  $\varphi = (\lambda v\psi)$ ,  $v \in VAR_{\sigma}$  and  $\psi \in ME_{\tau}$ , then  $[[\varphi]]^{M,g} =$   
The function  $p$  whose domain is  $D_{\sigma,E}$ , whose range is  $D_{\tau,E}$  and for all  $a \in D_{\sigma,E}$ ,  
 $p(a) = [[\psi]]^{M,g(v/a)}$

(29) **Illustration of the Semantics: A Model for TL**

Let the model  $\mathcal{M}$  be the pair  $\langle \{\text{Barack, Michelle, Mitt}\}, I \rangle$ , where  $I$  contains the following mappings (amongst infinitely many others):

- a.  $I(a_1) = \text{Barack}$
- b.  $I(a_2) = \text{Mitt}$
- c.  $I(a_3) = \text{Michelle}$
- d.  $I(Q_2) = h = \{ \langle \text{Michelle}, 0 \rangle, \langle \text{Barack}, 1 \rangle, \langle \text{Mitt}, 0 \rangle \}$
- e.  $I(Q_3) = i = \{ \langle \text{Michelle}, 0 \rangle, \langle \text{Barack}, 1 \rangle, \langle \text{Mitt}, 1 \rangle \}$
- f.  $I(Q_1) = k = \{ \langle \text{Michelle}, 0 \rangle, \langle \text{Barack}, 1 \rangle, \langle \text{Mitt}, 0 \rangle \}$
- e.  $f(R_3) = j = \begin{cases} \text{Michelle} & \rightarrow \left\{ \begin{array}{l} \text{Michelle} \rightarrow 1 \\ \text{Barack} \rightarrow 1 \\ \text{Mitt} \rightarrow 0 \end{array} \right\} \\ \text{Barack} & \rightarrow \left\{ \begin{array}{l} \text{Michelle} \rightarrow 1 \\ \text{Barack} \rightarrow 1 \\ \text{Mitt} \rightarrow 0 \end{array} \right\} \\ \text{Mitt} & \rightarrow \left\{ \begin{array}{l} \text{Michelle} \rightarrow 0 \\ \text{Barack} \rightarrow 0 \\ \text{Mitt} \rightarrow 1 \end{array} \right\} \end{cases}$

(30) **Illustration of the Semantics: Interpretation of Illustrative Expressions**

Let  $\mathcal{M}$  be the model defined in (29). Let  $g$  be some arbitrary variable assignment based on  $\mathcal{M}$ .

a.  $[(\lambda P_4 (P_4 a_2))]^{M,g} =$

The function  $p$  with domain  $D_{\langle e \rangle, E}$ , range  $D_{t,E}$  and for all  $a \in D_{\langle e \rangle, E}$ ,  
 $p(a) = a(\text{Mitt})$

b.  $[(\lambda P_4 \forall x_3 ((Q_3 x_3) \rightarrow (P_4 x_3)))]^{M,g} =$

The function  $p$  with domain  $D_{\langle e \rangle, E}$ , range  $D_{t,E}$  and for all  $a \in D_{\langle e \rangle, E}$ ,  
 $p(a) = 1$  iff for all  $a' \in D_{e,E}$ , either  $i(a') = 0$  or  $a(a') = 1$  =

The function  $p$  with domain  $D_{\langle e \rangle, E}$ , range  $D_{t,E}$  and for all  $a \in D_{\langle e \rangle, E}$ ,  
 $p(a) = 1$  iff for all  $a' \in D_{e,E}$ , if  $i(a') = 1$  then  $a(a') = 1$  =

**The characteristic function of the set of ‘properties every man has’**

---

3. **The Translation from ME+Q to TL: PTQ-Style Presentation**

In PTQ, the system for translating from ME+Q to TL differs slightly from that in UG.

- As we will see, however, the differences are not that deep or fundamental...

(31) **First Ingredient: Category-to-Type Mapping**

Just like with the UG notion of a translation base, one of the key ingredients to the PTQ translation system is a mapping from  $Cat$  to the set of types (in our logical language).

$f(TV) = <e, <e,t>>$	$f(IV) = <e,t>$
$f(S) = t$	$f(T) = <<e,t>, t>$
$f(CN) = <e,t>$	$f(PR) = e$

Note: The mapping above is *not at all* the one that actually appears in PTQ.

Recall that we’re right now just ‘converting’ our system from the last handout into the PTQ style.

(32) **Second Ingredient: Lexical Translation Function**

Again, just as in UG’s notion of a translation base, the second key ingredient to the PTQ translation system is a function mapping the basic expressions of ME+Q to ones in TL.

Let  $g$  be a function whose domain is the set of basic expressions in (5) **except for  $B_T$  and  $B_{PR}$** , and for all  $A \in Cat$ ,  $\alpha \in B_A$ , and  $\alpha \in \text{Domain}(g)$ ,  $g(\alpha) \in \text{CON}_{f(A)}$

- That is,  $g$  maps expressions of category  $A$  to *constants* of the corresponding type
- This restriction that  $g$  only maps to *constants* is unique to PTQ

### (33) Meta-Language Abbreviations

Recall that the constants of our language TL are all of the form  $c_{\tau, n}$ . In what follows, we'll make use of the following meta-language abbreviations.

- a. **loves'** = g(**loves**) [whatever  $\langle e, \langle e, t \rangle \rangle$  constant g maps **loves** to]
- b. **smokes'** = g(**smokes**) [whatever  $\langle e, t \rangle$  constant g maps **smokes** to]
- c. **man'** = g(**man**) [whatever  $\langle e, t \rangle$  constant g maps **man** to]
- d. **president'** = g(**president**) [whatever  $\langle e, t \rangle$  constant g maps **president** to]
- e. **woman'** = g(**woman**) [whatever  $\langle e, t \rangle$  constant g maps **woman** to]

### (34) Third Ingredient: Translation Rules

In the PTQ-system, the work done in UG by the translation base and the definition of the polynomial operations is instead done by a system of ‘translation rules’.

- These ‘translation rules’ stand in a one-to-one correspondence with the syntactic rules in (9)-(14)
  - Much as how in UG the polynomial operations are in correspondence with the syntactic operations, and the derived syntactic rules with the syntactic rules.
- As we'll see, these ‘translation rules’ do the work of both (i) defining the ‘polynomial operation’ each syntactic operation corresponds to; (ii) putting the operations in correspondence.

### (35) The Rule T1

- a. The Rule: Rule T1 consists of a collection of sub-rules.
  - (i) If  $\alpha$  is in the domain of g, then  $\alpha$  translates to  $g(\alpha)$ .
  - (ii) **Barack translates to**  $(\lambda P_4 (P_4 \text{ barack}'))$ , where **barack'**  $\in CON_e$   
**Mitt translates to**  $(\lambda P_4 (P_4 \text{ mitt}'))$ , where **mitt'**  $\in CON_e$   
**Michelle translates to**  $(\lambda P_4 (P_4 \text{ michelle}'))$ , where **michelle'**  $\in CON_e$
  - (iii) **he n and she n translate to:**  $x_n (= v_{e,n})$
- b. Remarks:
  - This ‘translation rule’ incorporates the following information:
    - (i) The lexical translation function g is a subset of the full translation relation
    - (ii) Proper names translate as  $\langle e, t \rangle$  formula; pronouns as type-e variables.
  - In the UG system, the information in (ii)-(iii) is part of the lexical translation function; in PTQ, though, that function can only map to constants (and so we need to pack it in as a separate rule)

(36) **The Rule T2**

- a. The Rule: If  $\zeta \in P_{CN}$  and  $\zeta$  translates to  $\zeta'$ , then
  - (i)  $F_0(\zeta)$  translates to  $(\lambda P_0 \forall x_0 ((\zeta' x_0) \rightarrow (P_0 x_0)))$
  - (ii)  $F_2(\zeta)$  translates to  $(\lambda P_0 \exists x_0 ((\zeta' x_0) \& (P_0 x_0)))$ ,
- b. Remarks: As mentioned in (34), rule T2 does both the following:
  - (i) Defines the ‘polynomial operations’ corresponding to  $F_0 (= H_{Every})$  and  $F_2 (= H_{Some})$
  - (ii) Puts these polynomial operations ‘in correspondence’ with  $F_0$  and  $F_2$  in the translation relation.

(37) **The Rule T4**

- a. The Rule:  
If  $\delta \in P_{PR}$  and  $\beta \in P_{IV}$ , and  $\delta, \beta$  translate to  $\delta', \beta'$  respectively, then  $F_4(\delta, \beta)$  translates to  $(\delta' \beta')$
- b. Remarks: Again, as mentioned in (34), rule T4 does both the following:
  - (i) Defines the ‘polynomial operation’ corresponding to  $F_4 (= H_{Merge-S})$
  - (ii) Puts that polynomial operation in correspondence with  $F_4$  in the translation relation.

(38) **The Rule T5**

If  $\delta \in P_{TV}$  and  $\beta \in P_{PR}$ , and  $\delta, \beta$  translate to  $\delta', \beta'$  respectively, then  $F_5(\delta, \beta)$  translates to  $(\delta' \beta')$ .

(39) **The Rule T11**

If  $\varphi, \psi \in P_S$ , and  $\varphi, \psi$  translate to  $\varphi', \psi'$  respectively, then  $F_8(\varphi, \psi)$  translates to  $(\varphi' \& \psi')$  and  $F_9(\varphi, \psi)$  translates to  $(\varphi' \vee \psi')$ .

(40) **The Rule T14**

- a. The Rule:  
If  $\alpha \in P_T$  and  $\varphi \in P_S$ , and  $\alpha, \varphi$  translate to  $\alpha', \varphi'$  respectively, then  $F_{10,n}(\alpha, \varphi)$  translates to  $(\alpha' (\lambda x_n \varphi'))$
- b. Remarks:
  - As in the corresponding syntactic rule S14, this translation rule covers not simply *one* syntactic operation, but a whole infinite family of them.
  - This translation rule implicitly does the work of defining an infinite family of polynomial operations  $H_{10,n}$ , each corresponding to  $F_{10,n}$

(41) **The Translation Relation**

The translation relation translates to between expressions of ME+Q and those of TL is the smallest binary relation satisfying T1-T14.

*Given the correspondence between the syntactic rules S1-S14 and the translation rules T1-T14, we can build up the translation for a sentence (rule-by-rule) as we construct it.*

(42) **Illustration of the Translation Rules, Part 1**

- (i) **man, loves, president** translate to **man'**, **loves'**, **president'** respectively ((33), T1)
- (ii) **he 1** and **he 2** translate to  $x_1$  and  $x_2$  respectively (T1)
- (iii)  $F_5(\text{loves}, \text{he } 2)$  translates to  $(\text{loves}' x_2)$  (T5)
- (iv) **loves him 2** translates to  $(\text{loves}' x_2)$  (def. of  $F_5$ )
- (v)  $F_4(\text{he } 1, \text{loves him } 2)$  translates to  $((\text{loves}' x_2) x_1)$  (T4)
- (vi) **he 1 loves him 2** translates to  $((\text{loves}' x_2) x_1)$  (def. of  $F_4$ )
- (vii)  $F_2(\text{man})$  translates to  $(\lambda P_0 \exists x_0 ((\text{man}' x_0) \& (P_0 x_0)))$  (T2)
- (viii) **some man** translates to  $(\lambda P_0 \exists x_0 ((\text{man}' x_0) \& (P_0 x_0)))$  (def. of  $F_2$ )
- (ix)  $F_{10,1}(\text{some man}, \text{he } 1 \text{ loves him } 2)$  translates to  $((\lambda P_0 \exists x_0 ((\text{man}' x_0) \& (P_0 x_0))) (\lambda x_1 ((\text{loves}' x_2) x_1)))$  (T14)
- (x) **some man loves him 2** translates to  $((\lambda P_0 \exists x_0 ((\text{man}' x_0) \& (P_0 x_0))) (\lambda x_1 ((\text{loves}' x_2) x_1)))$  (def. of  $F_{10,1}$ )
- (xi)  $F_0(\text{president})$  translates to  $(\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0)))$  (T2)
- (xii) **every president** translates to  $(\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0)))$  (def. of  $F_0$ )
- (xiii)  $F_{10,2}(\text{every president}, \text{some man loves him } 2)$  translates to  $((\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))) (\lambda x_2 ((\lambda P_0 \exists x_0 ((\text{man}' x_0) \& (P_0 x_0))) (\lambda x_1 ((\text{loves}' x_2) x_1)))))$  (T14)
- (xiv) **some man loves every president** translates to  $((\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))) (\lambda x_2 ((\lambda P_0 \exists x_0 ((\text{man}' x_0) \& (P_0 x_0))) (\lambda x_1 ((\text{loves}' x_2) x_1))))) \Leftrightarrow (\alpha\text{- and } \lambda\text{-conv.})$
- (xv)  $\forall x_0 ((\text{president}' x_0) \rightarrow \exists x_2 ((\text{man}' x_2) \& ((\text{loves}' x_0) x_2)))$

(43) **Illustration of the Translation Rules, Part 2**

Steps (i)-(vi) are exactly the same as those in (42):

- (vii)  $F_0(\text{president}) \text{ translates to } (\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0)))$  (T2)
- (viii) **every president** translates to  $(\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0)))$  (def. of  $F_0$ )
- (ix)  $F_{10,2}(\text{every president, he 1 loves him 2}) \text{ translates to}$   
 $((\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))) (\lambda x_2 ((\text{loves}' x_2) x_1)))$  (T14)
- (x) **he 1 loves every president** translates to  
 $((\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))) (\lambda x_2 ((\text{loves}' x_2) x_1)))$  (def. of  $F_{10,2}$ )
- (xi)  $F_2(\text{man}) \text{ translates to } (\lambda P_0 \exists x_0 ((\text{man}' x_0) \& (P_0 x_0)))$  (T2)
- (xii) **some man** translates to  $(\lambda P_0 \exists x_0 ((\text{man}' x_0) \& (P_0 x_0)))$  (def. of  $F_2$ )
- (xiii)  $F_{10,1}(\text{some man, he 1 loves every president}) \text{ translates to:}$   
 $((\lambda P_0 \exists x_0 ((\text{man}' x_0) \& (P_0 x_0)))$   
 $(\lambda x_1 ((\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))) (\lambda x_2 ((\text{loves}' x_2) x_1)))))$  (T14)
- (xiv) **some man loves every president** translates to:  
 $((\lambda P_0 \exists x_0 ((\text{man}' x_0) \& (P_0 x_0)))$   
 $(\lambda x_1 ((\lambda P_0 \forall x_0 ((\text{president}' x_0) \rightarrow (P_0 x_0))) (\lambda x_2 ((\text{loves}' x_2) x_1)))) \Leftrightarrow (\alpha\text{- and } \lambda\text{-conv.})$
- (xv)  $\exists x_0 ((\text{man}' x_0) \& \forall x_2 ((\text{president}' x_2) \rightarrow ((\text{loves}' x_2) x_0)))$

(44) **Key Observation**

As shown by (42) and (43), the relation ‘translates to’ is **not** a function.

- When our syntactic derivation of **some man loves every president** follows the procedure in Analysis Tree 1 in (17c), the translation is logically equivalent to:  
 $\forall x_0 ((\text{president}' x_0) \rightarrow \exists x_2 ((\text{man}' x_2) \& ((\text{loves}' x_2) x_0)))$
- When our syntactic derivation of **some man loves every president** follows the procedure in Analysis Tree 2 in (17c), the translation is logically equivalent to:  
 $\exists x_0 ((\text{man}' x_0) \& \forall x_2 ((\text{president}' x_2) \rightarrow ((\text{loves}' x_2) x_0)))$

That is, syntactically ambiguous strings in ME+Q can be paired with *more than one translation*, corresponding to the different ways the strings can be derived in the syntax.

- Of course, if the language were *not* syntactically ambiguous, then the translation relation would be a function....

#### 4. Introducing ‘Meaning Postulates’

##### (45) A Potential Issue for Our System

Our semantics for TL set up in Section 2 was quite general and permissive. Consequently, there are models for TL that would not be appropriate as (induced) interpretations of ME+Q.

Illustration:

- It is perfectly consistent with the definitions in Section 2 for a model of TL to map the constants **man'** and **woman'** to the same, or overlapping  $\langle \text{et} \rangle$ -functions.
- However, it is (maybe) part of the grammar of English that these terms are antonyms.

##### (46) Relevant Quote from Montague

“The interpretations of intensional logic may, by way of the translation relation, be made to play a second role as interpretations of English. Not all interpretations of intensional logic, however, would be reasonable candidates for interpretations of English. In particular, it would be reasonable in this context to restrict attention to those interpretations of intensional logic in which the following formulas are true...”  
(Montague 1974; p. 27)

##### (47) The Solution to Issue (45)

A ‘logically permissible model of TL’ is one in which formulae (47a,b) are true:

- a.  $\forall x_0 ((\text{man}' x_0) \rightarrow \sim(\text{woman}' x_0))$
- b.  $\forall x_0 ((\text{woman}' x_0) \rightarrow \sim(\text{man}' x_0))$

In our analysis of English, we only consider ‘logically permissible models of TL’.

##### (48) On ‘Meaning Postulates’

- The term ‘meaning postulate’ is often used (though not by Montague himself) to refer to such conditions on models for a language.
- Though the example in (47) is dubious,<sup>1</sup> it’s possible to imagine more plausible cases:
  - $\forall x_0 ((\text{bachelor}' x_0) \rightarrow (\text{man}' x_0))$
  - $\forall x_0 ((\text{wine}' x_0) \rightarrow \exists x_1 ((\text{grape}' x_1) \& ((\text{made-from}' x_1 x_0)))$
- In general, such ‘meaning postulates’ allow us to encode facts about lexical semantics into our overall analysis of the natural language.
  - Whether this is a good *theory* of lexical semantics, though, is up for debate...

<sup>1</sup> After all, one could argue that is simply a *contingent biological fact* that no men are women, rather than it being a part of the *lexical semantics* of ‘man’ and ‘woman’.

## 5. Direct and Indirect Interpretation in the PTQ-Style System

As should be obvious by now, given our model in (29), our translation system from Section 3 allows us to ‘indirectly interpret’ strings of ME+Q.

### (49) The Interpretation of ME+Q Relative to a Model and Variable Assignment

Let  $\varphi$  be a meaningful expression of ME+Q, and let  $\varphi$  translate to  $\varphi'$ . Let  $\mathcal{M}$  be a (logically permissible) model for TL and  $g$  be a variable assignment based on  $\mathcal{M}$ . We say that  $[[\varphi']]^{M,g}$  is the **interpretation of  $\varphi$  under translation  $\varphi'$**  relative to  $\mathcal{M}$  and  $g$ .

Illustration:

- The interpretation of **some man loves every president** under translation (42) and relative to the model in (29) is 1
  - After all, Barack is the only president, and he loves himself.
- The interpretation of **some man loves every president** under translation (43) and relative to the model in (29) is also 1
  - Again, this is rendered true by the fact that Barack loves himself.

### (50) The Equivalence of Indirect and Direct Interpretation in UG

Once a translation base  $T$  is specified for  $L$  and  $L'$ , if  $L'$  has an interpretation  $B'$ , it is trivial (mechanical) to specify an interpretation  $B$  for  $L$ .

- (51) a. Question: Is indirect interpretation similarly ‘eliminable’ in a PTQ-style system?  
b. Answer:  
Yes, but it is a little less trivial/mechanical to build the model  $\mathcal{M}$  for natural language  $L$  on the basis of model  $\mathcal{M}$  for logical language  $L'$  and the translation.

### (52) A General Method for Recasting Indirect Interpretation as Direct Interpretation

For each ‘translation rule’  $T_n$ , specify a new ‘interpretation rule’  $I_n$ , where the output of  $I_n$  is the model-theoretic interpretation of  $T_n$

(53) **Direct Interpretation of ME+Q, Step 1: Defining the Models**

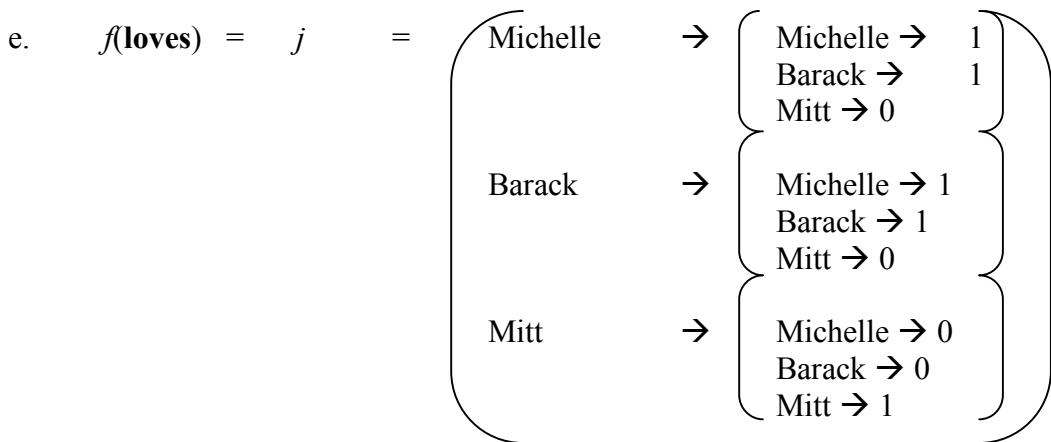
A model  $\mathcal{M}$  for ME+Q is a pair  $\langle E, I \rangle$  consisting of (i) a *non-empty* set  $E$ , called the ‘domain of  $\mathcal{M}$ ', and (ii) a function  $I$ , whose domain is (a) and whose range satisfies the condition in (b).

- (a) *Domain of I:*  $B_{TV} \cup B_{IV} \cup B_{CN}$
- (b) *Condition on Range of I:*
  1. If  $\varphi \in B_{TV}$ , then  $I(\varphi) \in D_{\langle e, \langle e, t \rangle \rangle, E}$
  2. If  $\varphi \in B_{IV}$ , then  $I(\varphi) \in D_{\langle e, t \rangle, E}$
  3. If  $\varphi \in B_{CN}$ , then  $I(\varphi) \in D_{\langle e, t \rangle, E}$

(54) **Illustration: A Model for ME+Q**

Let the model  $\mathcal{M}$  be the pair  $\langle \{\text{Barack, Michelle, Mitt}\}, I \rangle$ , where  $I$  contains the following mappings (amongst infinitely many others):

- a.  $I(\text{smokes}) = h = \{ \langle \text{Michelle}, 0 \rangle, \langle \text{Barack}, 1 \rangle, \langle \text{Mitt}, 0 \rangle \}$
- b.  $I(\text{man}) = i = \{ \langle \text{Michelle}, 0 \rangle, \langle \text{Barack}, 1 \rangle, \langle \text{Mitt}, 1 \rangle \}$
- c.  $I(\text{president}) = k = \{ \langle \text{Michelle}, 0 \rangle, \langle \text{Barack}, 1 \rangle, \langle \text{Mitt}, 0 \rangle \}$
- d.  $I(\text{woman}) = l = \{ \langle \text{Michelle}, 1 \rangle, \langle \text{Barack}, 0 \rangle, \langle \text{Mitt}, 0 \rangle \}$



(55) **Direct Interpretation of ME+Q, Step 2: Defining Variable Assignment**

Let  $\mathcal{M}$  be a model  $\langle E, I \rangle$  for ME+Q. Then  $g$  is a *variable assignment* (*based on  $\mathcal{M}$* ) if its domain is equal to (i) and its range satisfies the property in (ii).

- (i) *Domain of g:*  $\mathbb{N}$
- (iv) *Condition on Range of g:* For all  $n \in \mathbb{N}$ ,  $g(n) \in E$

We now must lay out a definition for ‘interpretation relative to a model and a variable assignment’...

Notice how the definition below mirrors our translation rules T1-T14

- (56) **Direct Interpretation of ME+Q, Step 3: Interpretation w.r.t. Model and Assignment**  
 Let  $\mathcal{M}$  be a model  $\langle E, I \rangle$  for ME+Q and  $g$  be a variable assignment based on  $\mathcal{M}$ . The relation ‘ $X$  is an interpretation of  $\varphi$  relative to  $\mathcal{M}$  and  $g$ ’  $[[\cdot]]^{M,g}$  is defined as follows:

- a. **Rule I1:**
  - (i) If  $\varphi$  is in the domain of  $I$ , then  $[[\varphi]]^{M,g} = I(\varphi)$
  - (ii)  $[[\text{Barack}]]^{M,g} =$   
 the function  $p$  with domain  $D_{\langle e, t \rangle, E}$  such that for all  $f \in D_{\langle e, t \rangle, E}$   
 $p(f) = 1$  iff  $f(\beta) = 1$  (where  $\beta$  is some specified member of  $E$ )
  - $[[\text{Mitt}]]^{M,g} =$   
 the function  $p$  with domain  $D_{\langle e, t \rangle, E}$  such that for all  $f \in D_{\langle e, t \rangle, E}$   
 $p(f) = 1$  iff  $f(\mu) = 1$  (where  $\mu$  is some specified member of  $E$ )
  - $[[\text{Michelle}]]^{M,g} =$   
 the function  $p$  with domain  $D_{\langle e, t \rangle, E}$  such that for all  $f \in D_{\langle e, t \rangle, E}$   
 $p(f) = 1$  iff  $f(v) = 1$  (where  $v$  is some specified member of  $E$ )
  - (iii)  $[[\text{he } n]]^{M,g} = [[\text{she } n]]^{M,g} = g(n)$
- b. **Rule I2:**
  - (i) If  $\varphi \in P_{CN}$ , then  $[[F_0(\varphi)]]^{M,g} =$   
 the function  $p$  with domain  $D_{\langle e, t \rangle, E}$  such that for all  $f \in D_{\langle e, t \rangle, E}$   
 $p(f) = 1$  iff for all  $x \in E$ , if  $[[\varphi]]^{M,g}(x) = 1$ , then  $f(x) = 1$
  - (ii) If  $\varphi \in P_{CN}$ , then  $[[F_2(\varphi)]]^{M,g} =$   
 the function  $p$  with domain  $D_{\langle e, t \rangle, E}$  such that for all  $f \in D_{\langle e, t \rangle, E}$   
 $p(f) = 1$  iff there is an  $x \in E$  such that  $[[\varphi]]^{M,g}(x) = 1$  and  $f(x) = 1$
- c. **Rule I4:** If  $\delta \in P_{PR}$  and  $\beta \in P_{IV}$ , then  $[[F_4(\delta, \beta)]]^{M,g} = [[\beta]]^{M,g}([[[\delta]]^{M,g}])$
- d. **Rule I5:** If  $\delta \in P_{TV}$  and  $\beta \in P_{PR}$ , then  $[[F_5(\delta, \beta)]]^{M,g} = [[\delta]]^{M,g}([[[\beta]]^{M,g}])$
- e. **Rule I11:** If  $\varphi, \psi \in P_S$ , then  $[[F_8(\varphi, \psi)]]^{M,g} = 1$  iff  $[[\varphi]]^{M,g} = [[\psi]]^{M,g} = 1$ , and  
 $[[F_9(\varphi, \psi)]]^{M,g} = 1$  iff  $[[\varphi]]^{M,g} = 1$  or  $[[\psi]]^{M,g} = 1$
- f. **Rule I14:**  
 If  $\alpha \in P_T$  and  $\varphi \in P_S$ , then  $[[F_{10,n}(\alpha, \varphi)]]^{M,g} =$   
 $[[\alpha]]^{M,g}(\text{the function } p \text{ with domain } E \text{ such that for all } x \in E, p(x) = [[\varphi]]^{M,g(n/x)})$

We can now use the definitions in (53)-(55) to directly interpret expressions of ME+Q!!

(57) **Illustration: Directly Interpreting a Sentence of ME+Q**

Let  $\mathcal{M}$  be the model defined in (54) and  $g$  be any variable assignment based on  $\mathcal{M}$ .

- (i)  $[[ \text{every man smokes} ]]^{M,g} = \text{(by definition of ME+Q)}$
- (ii)  $[[ F_{10,1} ( F_0(\text{man}), F_4( \text{he 1}, \text{smokes} ) ) ]]^{M,g} = \text{(by I14)}$
- (iii)  $[[F_0(\text{man})]]^{M,g} (\text{the function } p \text{ with domain E such that for all } x \in E, p(x) = [[F_4( \text{he 1}, \text{smokes} )]]^{M,g(1/x)}) = \text{(by I4)}$
- (iv)  $[[F_0(\text{man})]]^{M,g} (\text{the function } p \text{ with domain E such that for all } x \in E, p(x) = [[\text{smokes}]]^{M,g(n^1x)}([[\text{he 1}]]^{M,g(1/x)})) = \text{(by I1)}$
- (v)  $[[F_0(\text{man})]]^{M,g} (\text{the function } p \text{ with domain E such that for all } x \in E, p(x) = I(\text{smokes})(g(1/x)(1))) = \text{(by (54))}$
- (vi)  $[[F_0(\text{man})]]^{M,g} (\text{the function } p \text{ with domain E such that for all } x \in E, p(x) = h(x)) = \text{(by meta-logical reasoning)}$
- (vii)  $[[F_0(\text{man})]]^{M,g}(h) = \text{(by I2)}$
- (viii)  $(\text{the function } p \text{ with domain } D_{<e,t>,E} \text{ such that for all } f \in D_{<e,t>,E} \text{ iff for all } x \in E, \text{ if } [[\text{man}]]^{M,g}(x) = 1, \text{ then } f(x) = 1)(h) = \text{(by I1)}$
- (ix)  $(\text{the function } p \text{ with domain } D_{<e,t>,E} \text{ such that for all } f \in D_{<e,t>,E} \text{ iff for all } x \in E, \text{ if } i(x) = 1, \text{ then } f(x) = 1)(h) = \text{(by meta-logical reasoning)}$
- (x)  $1 \text{ iff for all } x \in E, \text{ if } i(x) = 1, \text{ then } h(x) = 1 = \text{(by (54))}$
- (xi)  $0$

(58) **Remark**

- Note that for syntactically ambiguous strings, there can be two possible interpretations w.r.t a model and variable assignment.
- Thus, as the language in (56) suggests,  $[[.]]^{M,g}$  is not a function.

(59) **What We've Done So Far**

- Reviewed the UG architecture, and shown how it can provide a compositional semantics for a fragment of English with quantificational ‘terms’ (NPs)
- Reviewed the PTQ architectures, and shown how it also can provide a compositional semantics for a fragment of English with quantificational ‘terms’ (NPs)
- Examined how the systems in UG and PTQ relate to one another...

Now, let's actually study Montague's full semantic analysis of English in PTQ...

## The Proper Treatment of Quantification in Ordinary English, Part 1: The Fragment of English

We will now explore the analysis of English that Montague puts forth in his seminal paper, PTQ. As we've already seen, there are three principle parts to the analysis:

- The syntactic fragment of English (easy)
- The logical language, Intensional Logic (moderate)
- The translation from English to IL (difficult; after all, this is the actual semantic analysis)

### (1) On UG vs. PTQ

"On their common domain of applicability, the three treatments [UG, PTQ, and *English as a Formal Language*] essentially agree in the truth and entailment conditions imposed on sentences... Nevertheless, the details of the present development possess certain aesthetic merits, of coherence and conceptual simplicity, not to be found in the treatment of English in [UG]."

## 1. The Syntactic Categories of English

### (1) The Categories

- In PTQ, Montague employs a system of syntactic categories (i.e., category *labels*) for English that are similar in structure to the types (i.e., a so-called categorical grammar)
- This allows for an elegant statement of the category-to-type correspondence between English and IL.

*Cat* is the smallest set such that:

- a.  $e, t \in Cat$
- b. If  $A, B \in Cat$ , then  $A/B, A//B \in Cat$

### (2) On Reading the Category Labels

From the point of view of our type notation, the category notation is 'flipped backwards'

$A/B$  and  $A//B$  = Expressions that when 'combined' with an expression of category B yield an expression of category A.

Illustration:

$t/e, t//e$ : Combines with an expression of category e to yield one of category t  
 $(\approx <e,t>)$

- (3) a. Question: What is the difference between A/B and A//B?

b. Answer: Nothing substantial.  
As we'll see in a second, it allows Montague to distinguish between CNs and IVs.

c. Quote:  
“We shall regard the categories A/B and A//B as playing the same semantical but different syntactical roles.”  
(*i.e.*, CN and IV have translations/meanings of the same type, but are different syntactic categories in English.)

#### (4) Abbreviations for the Syntactic Categories

- Up until now, we've been treating the labels below as the actual category labels of our language.
  - In PTQ, though, Montague introduces them as *meta-linguistic abbreviations* for the more complex category labels defined in (1).

a. Some New Definitions for Old Friends

(i)	IV	<i>abbreviates</i>	t/e	
(ii)	T	<i>abbreviates</i>	t/IV	( = t/(t/e))
(iii)	CN	<i>abbreviates</i>	t//e	
(iv)	TV	<i>abbreviates</i>	IV/T	( = (t/e)/(t/(t/e)) )

Note: TVs are expressions that combine with **Terms** ( $t/(t/e)$ ) to yield IVs ( $t/e$ ).  
*Thus, TVs will directly combine syntactically with quantificational terms...*

### b. Some New Friends

<u>Category</u>	<u>Informal Name</u>	<u>Abbreviates</u>	<u>Example</u>
(i)	IAV	'IV-modifying adverb'	IV/IV
(ii)	t/t	'Sentence-modifying adverb'	necessarily
(iii)	IAV/T	'IAV-making preposition'	in, about
(iv)	IV/t	'Sentence-taking verb phrase'	believe that
(v)	IV//IV	'IV-taking verb phrase'	try to, wish to

### *Notes:*

- Montague uses the term ‘verb phrase’ in (iv) and (v) a bit differently from syntacticians
  - Just like TVs, prepositions in PTQ directly combine with quantificational terms.
  - The only uses of the A/B and A//B distinction are:
    - IV vs. CN                    (*run* vs. *man*)
    - IAV vs. IV//IV            (*slowly* vs. *try to*)

(5) **Remark**

In PTQ, there are an infinite number of category labels for English, some of which don't seem to apply to any actual expression of English (e.g. e/t)

- Again, what this buys us is mainly just an elegant statement of the type-category correspondence in the translation system...

---

## 2. The Basic Expressions of English

*The following are the basic expressions of the English fragment, exactly as written out by Montague in PTQ.*

(6) **Basic Expressions of the English Fragment**

- a.  $B_{IV} = \{\mathbf{run, walk, talk, rise, change}\}$
- b.  $B_T = \{\mathbf{John, Mary, Bill, ninety, he_0, he_1, he_2, ...}\}$
- c.  $B_{TV} = \{\mathbf{find, lose, eat, love, date, be, seek, conceive}\}$
- d.  $B_{IAV} = \{\mathbf{rapidly, slowly, voluntarily, allegedly}\}$
- e.  $B_{CN} = \{\mathbf{man, woman, park, fish, pen, unicorn, price, temperature}\}$
- f.  $B_{t/t} = \{\mathbf{necessarily}\}$
- g.  $B_{IAV/T} = \{\mathbf{in, about}\}$
- h.  $B_{IV/t} = \{\mathbf{believe that, assert that}\}$
- i.  $B_{IV//IV} = \{\mathbf{try to, wish to}\}$
- j.  $B_A = \emptyset$  if A is any category other than those mentioned above. (In particular, the sets  $B_e$  of basic entity expressions and  $B_t$  of basic declarative sentences are empty.)<sup>1</sup>

(7) **Remark**

In the PTQ system, pronouns are all terms. They are also all masculine. Finally, note that the pronouns and indices are not syntactically separate expressions (unlike in UG).

---

<sup>1</sup> Just to forestall any confusion, in the PTQ paper itself, Montague uses 'A' to denote the null set.

(8) **Key Observation, Previewing Some Fun to Come...**

- Note that the following pairs of expressions are all members of the same category:

a.	eat, seek	[both TVs]
b.	rapidly, allegedly	[both IAVs]
c.	in, about	[both IAV/Ts]
- In each of these pairs, the second expression creates an ‘opaque’ environment, whereas the first one does not (i.e., the first creates a ‘transparent’ environment)

a.	(i) John ate a unicorn.	(entails ‘there is a unicorn’)
	(ii) John seeks a unicorn.	(doesn’t entail ‘there is a unicorn’)
b.	(i) John rapidly danced.	(entails ‘John danced’)
	(ii) John allegedly danced.	(doesn’t entail ‘John danced’)
c.	(i) John talked in a house.	(entails ‘there is a house’)
	(ii) John talked about a unicorn.	(doesn’t entail ‘there is a unicorn’)
- Given that *seek*, *allegedly*, and *about* create ‘opaque’ contexts, we’d ideally want them to take *intensions* as arguments (LING 620).
  - *Seek* takes as argument the intension of *a unicorn*.
  - *Allegedly* takes as argument the intension of *danced*.
  - *About* takes as argument the intension of *a unicorn*.
- **Recall, however, Montague’s requirement that expressions of the same category map to translations/meanings of the same type.**
- Consequently, since *seek*, *allegedly*, *about* have a meaning that takes intensions as arguments, **we’ll also need for *eat*, *rapidly*, and *in* to have such meanings...**

---

3. **The Syntactic Rules**

Given the basic expressions in (6), the syntactic rules outlined in this section will simultaneously define the set of meaningful expressions of English,  $\bigcup_{A \in Cat} P_A$

(9) **Rule S1**       $B_A \subseteq P_A$ , for every category A.

(10) **Rule S2 (Forming Quantificational Terms)**

If  $\zeta \in P_{CN}$ , then  $F_0(\zeta), F_1(\zeta), F_2(\zeta) \in P_T$ ,

where  $F_0(\zeta) = \text{every } \zeta$

$F_1(\zeta) = \text{the } \zeta$

$F_2(\zeta)$  is **a**  $\zeta$  or **an**  $\zeta$  according as the first word in  $\zeta$  takes **a** or **an**

### (11) Remarks on Rule S2

- a. Via the addition of  $F_1$ , we are adding definite determiners to our fragment. Montague will provide a ‘Russellian’ analysis of definite terms:<sup>2</sup>

*The man smokes* is true iff  $\exists x_0 \forall x_1 (((\text{man}' x_1) \leftrightarrow x_0 = x_1) \& (\text{smokes}' x_0))$   
*There is exactly one man, and he smokes.*

- b. The definition of  $F_2$  in (10) appeals to a notion that Montague does not ever explicitly define in the paper: whether a given word takes **a** or **an**
  - In this case, though, we could easily revise (10) to explicitly contain the generalization ‘**an** before a vowel’.

### (12) Rule S3 (Relativization)

If  $\zeta \in P_{CN}$  and  $\varphi \in P_t$ , then  $F_{3,n}(\zeta, \varphi) \in P_{CN}$ , where  $F_{3,n}(\zeta, \varphi) = \zeta$  such that  $\varphi'$ , and  $\varphi'$  comes from  $\varphi$  by replacing each occurrence of **he<sub>n</sub>** or **him<sub>n</sub>** by {**he**, **she**, **it**} or {**him**, **her**, **it**}, respectively, according as the first  $B_{CN}$  in  $\zeta$  is of {masc., fem., neuter} gender.

Illustration:

- |   |                      |
|---|----------------------|
| (i) <b>woman</b> $\in P_{CN}$ , <b>John loves him<sub>5</sub></b> $\in P_t$ | (Rules)              |
| (ii) $F_{3,5}(\text{woman}, \text{John loves him}_5) \in P_{CN}$            | (Rule S3)            |
| (iii) <b>woman such that John loves her</b> $\in P_{CN}$                    | (def. of $F_{3,5}$ ) |

### (13) Remarks on Rule S3

- a. Rule S3 is the rule for forming relative clauses in PTQ. Note that it only forms ‘such that’ relatives; there are no mechanisms in PTQ for filler-gap dependencies.
- b. Like  $F_{10,n}$  from the last handout,  $F_{3,n}$  is an infinite family of operations.
- c. Also like  $F_{10,n}$ , the translation operation corresponding to  $F_{3,n}$  will lambda abstract over the variable with index  $n$ .

*Rough, Simplified Illustration:*

$F_{3,5}(\text{woman}, \text{John loves him}_5)$  translates to:  $(\lambda x_5 ((\text{woman}' x_5) \& ((\text{loves } x_5) \text{john}')))$   
**woman such that John loves her** translates to:  $(\lambda x_5 ((\text{woman}' x_5) \& ((\text{loves } x_5) \text{john}')))$

---

<sup>2</sup> Note that, since Intensional Logic is being used as the translation language, it would not be possible in the PTQ system to provide a Fregean/presuppositional analysis of definite terms.

(14) **Rule S4 (Subject-Predicate Rule)**

If  $\alpha \in P_{t/IV}$  and  $\delta \in P_{IV}$ , then  $F_4(\alpha, \delta) \in P_t$ , where  $F_4(\alpha, \delta) = \alpha \ \delta'$  and  $\delta'$  is the result of replacing the first *verb* (i.e., member of  $B_{IV}$ ,  $B_{TV}$ ,  $B_{IV/t}$ , or  $B_{IV//IV}$ ) in  $\delta$  by its third person singular present.

(15) **Remarks on Rule S4**

- a. In PTQ, the operation  $F_4$  doesn't just combine a subject with a predicate, it is also responsible for adding the tense and agreement morphology to the verb.
- b. Due to its additional morpho-syntactic role, the definition of  $F_4$  appeals to two important, fundamental concepts: (a) the notion of a 'verb', and (b) the notion of a verb's 'third person singular present'.
- c. The notion 'verb' is defined as part of the definition of  $F_4$ . *Notice, though, that 'verb' is not actually a syntactic category in the system.*
  - In fact, *verb* can't be a syntactic category in MG, given the need for category-to-type correspondence.
- d. The notion 'third person singular present' is nowhere defined in the paper. To properly implement this, though, we could imagine defining a function '3sgPRES' which maps an English verb root to its 3rd singular present form:

$$\begin{aligned} 3\text{sgPRES}(root) &= \begin{cases} \text{does,} & \text{if } root = \text{do} \\ \text{is,} & \text{if } root = \text{be} \\ \text{has,} & \text{if } root = \text{have} \\ \dots & \\ root + \text{s} & \text{otherwise} \end{cases} \end{aligned}$$

- With this function, we'd simply say that  $\delta'$  is obtained from  $\delta$  by replacing the first verb  $v$  in  $\delta$  with  $3\text{sgPRES}(v)$ .

(15) **Rule S5 (Direct Object Rule)**

If  $\delta \in P_{IV/T}$  and  $\beta \in P_T$ , then  $F_5(\delta, \beta) \in P_{IV}$ , where  $F_5(\delta, \beta) = \delta \ \beta$  if  $\beta$  does not have the form  $\mathbf{he}_n$ , and  $F_5(\delta, \mathbf{he}_n) = \delta \ \mathbf{him}_n$

Remark:

Again, the definition of  $F_5$  captures the behavior of objective case on English pronouns.

(16) **Rule S6 (Prepositional Phrase Rule)**      If  $\delta \in P_{IAV/T}$  and  $\beta \in P_t$ , then  $F_5(\delta, \beta) \in P_{IAV}$

- Illustration:
- |       |   |                  |
|-------|---|------------------|
| (i)   | <b>in</b> $\in P_{IAV/T}$ , <b>he<sub>2</sub></b> $\in P_t$ | (Rule S1)        |
| (ii)  | $F_5(\text{in}, \text{he}_2) \in P_{IAV}$                   | (Rule S6)        |
| (iii) | <b>in him<sub>2</sub></b> $\in P_{IAV}$                     | (def. of $F_5$ ) |

Remark:      S6 forms prepositional phrases in PTQ. Note its use of the operation  $F_5$

- o This captures the presence of ACC on pronominal complements of Ps
- o **This also nicely illustrates the difference between syntactic operations and syntactic rules in PTQ**

(17) **Rule S7 (Finite Complement Clause Rule)**

If  $\delta \in P_{IV/t}$  and  $\beta \in P_t$ , then  $F_6(\delta, \beta) \in P_{IV}$ , where  $F_6(\delta, \beta) = \delta \beta$

Illustration:

- |       |   |                  |
|-------|---|------------------|
| (i)   | <b>believe that</b> $\in P_{IV/t}$ , <b>John runs</b> $\in P_t$ | (Rule S1, S4)    |
| (ii)  | $F_6(\text{believe that}, \text{John runs}) \in P_{IV}$         | (Rule S7)        |
| (iii) | <b>believe that John runs</b> $\in P_{IV}$                      | (def. of $F_6$ ) |

Question:      Did Montague really need to introduce a new operation  $F_6$  here?  
 Couldn't he have simply continued to use  $F_5$ ?  
 (After all, nothing in  $P_t$  will be of the form **he<sub>n</sub>**)

(18) **Rule S8 (Infinitival Control Rule)**

If  $\delta \in P_{IV//IV}$  and  $\beta \in P_{IV}$ , then  $F_6(\delta, \beta) \in P_{IV}$

Illustration:

- |       |   |                  |
|-------|---|------------------|
| (i)   | <b>try to</b> $\in P_{IV//IV}$ , <b>find a unicorn</b> $\in P_{IV}$ | (Rule S1, S5)    |
| (ii)  | $F_6(\text{try to}, \text{find a unciorn}) \in P_{IV}$              | (Rule S8)        |
| (iii) | <b>try to find a unicorn</b> $\in P_{IV}$                           | (def. of $F_6$ ) |

(19) **Rule S9 (Sentential Adverbs Rule)**

If  $\delta \in P_{t/t}$  and  $\beta \in P_t$ , then  $F_6(\delta, \beta) \in P_t$

Illustration:

- |       |   |                  |
|-------|---|------------------|
| (i)   | <b>necessarily</b> $\in P_{t/t}$ , <b>John runs</b> $\in P_t$ | (Rule S1, S4)    |
| (ii)  | $F_6(\text{necessarily}, \text{John runs}) \in P_t$           | (Rule S9)        |
| (iii) | <b>necessarily John runs</b> $\in P_t$                        | (def. of $F_6$ ) |

(20) **Rule S10 (Adverbs Rule)**

If  $\delta \in P_{IV/IV}$  and  $\beta \in P_{IV}$ , then  $F_7(\delta, \beta) \in P_{IV}$ , where  $F_7(\delta, \beta) = \beta \delta$

Illustration:

- |       |   |                  |
|-------|---|------------------|
| (i)   | <b>about a unicorn</b> $\in P_{IV/IV}$ , <b>talk</b> $\in P_{IV}$ | (Rule S1, S6)    |
| (ii)  | $F_7(\text{about a unicorn}, \text{talk}) \in P_{IV}$             | (Rule S10)       |
| (iii) | <b>talk about a unicorn</b> $\in P_{IV}$                          | (def. of $F_7$ ) |

(21) **Rule S11 (Sentential Conjunction/Disjunction Rule)**

If  $\varphi, \psi \in P_t$ , then  $F_8(\varphi, \psi), F_9(\varphi, \psi) \in P_t$ , where  $F_8(\varphi, \psi) = \varphi \text{ and } \psi$  and  $F_9(\varphi, \psi) = \varphi \text{ or } \psi$ .

(22) **Rule S12 (IV Conjunction/Disjunction Rule)**

If  $\varphi, \psi \in P_{IV}$ , then  $F_8(\varphi, \psi), F_9(\varphi, \psi) \in P_{IV}$

Illustration:

- |  |                            |
|--|----------------------------|
| (i) <b>runs, loves Mary</b> $\in P_{IV}$                                     | (Rule S1, S5)              |
| (ii) $F_8(\text{runs, loves Mary}), F_9(\text{runs, loves Mary}) \in P_{IV}$ | (Rule S12)                 |
| (iii) <b>runs and loves Mary, runs or loves Mary</b> $\in P_{IV}$            | (def. of $F_8$ and $F_9$ ) |

(23) **Remark**

- Montague uses the *same* syntactic operations,  $F_8$  and  $F_9$ , to do conjunction/disjunction of sentences and conjunction/disjunction of IVs.
- As we'll see later, this will cause some difficulty converting the PTQ system into the algebraic UG format
  - In the putative translation base, we'd need a *single* polynomial operation over IL to correspond to  $F_8$  ( $F_9$ )
  - But a single such operation won't give us the right translations for *both* sentential conjunction (disjunction) and IV conjunction (disjunction)

(24) **Rule S13 (Term Disjunction)**      If  $\alpha, \beta \in P_T$ , then  $F_9(\alpha, \beta) \in P_T$

Illustration:

- |  |                  |
|--|------------------|
| (i) <b>every man, John</b> $\in P_T$       | (Rule S1, S2)    |
| (ii) $F_9(\text{every man, John}) \in P_T$ | (Rule S13)       |
| (iii) <b>every man or John</b> $\in P_T$   | (def. of $F_9$ ) |

(25) a.    Question:      Why didn't Montague also include a rule for *conjoining* terms?

b.    Educated Guess:

He didn't want to have to deal with the collective reading of sentences like *John and Mary ate a unicorn*.

*And, at last we come to one of the centerpieces of the paper... the rules for 'Quantifying In'...*

(26) **Rule S14 (Quantifying In to Sentences)**

If  $\alpha \in P_T$  and  $\varphi \in P_t$ , then  $F_{10,n}(\alpha, \varphi) \in P_t$ , where either:

- (i)  $\alpha$  does not have the form **he<sub>k</sub>** and  $F_{10,n}(\alpha, \varphi)$  comes from  $\varphi$  by replacing the first occurrence of **he<sub>n</sub>** or **him<sub>n</sub>** by  $\alpha$ , and all other occurrences of **he<sub>n</sub>** or **him<sub>n</sub>** by {**he**, **she**, **it**} or {**him**, **her**, **it**} respectively, according as the gender of the first  $B_{CN}$  or  $B_T$  in  $\alpha$  is {masculine, feminine, neuter}, or
- (ii)  $\alpha = \text{he}_k$  and  $F_{10,n}(\alpha, \varphi)$  comes from  $\varphi$  by replacing all occurrences of **he<sub>n</sub>** or **him<sub>n</sub>** by **he<sub>k</sub>** or **him<sub>k</sub>** respectively.

(27) **Remark 1**

The definition of  $F_{10,n}(\alpha, \varphi)$  is disjunctive; its value depends upon whether the ‘term argument’  $\alpha$  is a pronoun or not.

- a. If  $\alpha$  is *not* a pronoun, then we do the following:

- (i) Replace the first instance of **he<sub>n</sub>** or **him<sub>n</sub>** with  $\alpha$ , and
- (ii) All subsequent instances of **he<sub>n</sub>** or **him<sub>n</sub>** with a pronoun agreeing in gender with  $\alpha$

Illustration:

- (i) **a woman**  $\in P_T$ , **he<sub>2</sub> runs and John likes him<sub>2</sub>**  $\in P_t$  (Rules)
- (ii)  $F_{10,2}(\text{a woman}, \text{he}_2 \text{ runs and John likes him}_2) \in P_t$  (Rule S14)
- (iii) **a woman runs and John likes her**  $\in P_t$  (def. of  $F_{10,n}$ )

- b. If  $\alpha$  is a pronoun, then we do the following:

Replace *all* instances of **he<sub>n</sub>** or **him<sub>n</sub>** with  $\alpha$  or its accusative variant, respectively

Illustration:

- (i) **he<sub>3</sub>**  $\in P_T$ , **he<sub>2</sub> runs and John likes him<sub>2</sub>**  $\in P_t$  (Rules)
- (ii)  $F_{10,2}(\text{he}_3, \text{he}_2 \text{ runs and John likes him}_2) \in P_t$  (Rule S14)
- (iii) **he<sub>3</sub> runs and John likes him<sub>3</sub>**  $\in P_t$  (def. of  $F_{10,n}$ )

- (28) a. **Question:** Why did Montague use this disjunctive definition of  $F_{10,n}$ ?  
After all, if we simply extended the first condition in (26ai) to pronouns, we’d end up getting almost the same strings:

*Illustration:* (ii)  $F_{10,2}(\text{he}_3, \text{he}_2 \text{ runs and John likes him}_2) \in P_t$   
(iii) **he<sub>3</sub> runs and John likes him**  $\in P_t$

- b. **Answer:**

For whatever reason, condition (26ai) doesn’t copy the index onto the subsequent pronouns. Consequently, if we quantified-in **a woman** on (28aiii), we wouldn’t get gender agreement on the following pronouns.

(29) **Remark 2**

The definition of  $F_{10,n}(\alpha, \varphi)$  again appeals to a notion that Montague doesn't ever define in the paper PTQ itself: the gender of a CN or T.

a. A Possible Implementation:

We could imagine defining a function GEN whose domain is  $B_{CN}$  and  $B_T$  and whose range is {MASC, FEM, NEUT}.

$$GEN(\mathbf{John}) = \text{MASC}$$

$$GEN(\mathbf{Mary}) = \text{FEM}$$

$$GEN(\mathbf{ninety}) = \text{NEUT}$$

...

$$GEN(\mathbf{man}) = \text{MASC}$$

$$GEN(\mathbf{woman}) = \text{FEM}$$

$$GEN(\mathbf{fish}) = \text{NEUT}$$

...

We could then reformulate (26) so that it makes reference to this GEN function.

(30) **Rule S15 (Quantifying In to NPs)** If  $\alpha \in P_T$  and  $\varphi \in P_{CN}$ , then  $F_{10,n}(\alpha, \varphi) \in P_{CN}$

Illustration:

- (i) **every man**  $\in P_T$ , **woman such that he<sub>2</sub> likes her**  $\in P_{CN}$  (Rules)
- (ii)  $F_{10,2}(\mathbf{every man}, \mathbf{woman such that he}_2 \mathbf{likes her}) \in P_{CN}$  (Rule S15)
- (iii) **woman such that every man likes her**  $\in P_{CN}$  (def. of  $F_{10,n}$ )

Remark: It still isn't clear to me what use this rule is.

Neither Montague himself nor Dowty et al. (1981) discuss it in detail.

(31) **Rule S16 (Quantifying In to VPs)** If  $\alpha \in P_T$  and  $\varphi \in P_{IV}$ , then  $F_{10,n}(\alpha, \varphi) \in P_{IV}$

Illustration:

- (i) **a unicorn**  $\in P_T$ , **find him<sub>2</sub> and eat him<sub>2</sub>**  $\in P_{IV}$  (Rules)
- (ii)  $F_{10,2}(\mathbf{a unicorn}, \mathbf{find him}_2 \mathbf{and eat him}_2) \in P_{IV}$  (Rule S16)
- (iii) **find a unicorn and eat it**  $\in P_{IV}$  (def. of  $F_{10,n}$ )

Remark: As we'll see, this allows us to capture the opaque/bound reading of "John wants to find a unicorn and eat it."

*Finally, to wrap things up, Montague introduces rules for adding negation and tense morphology...*

(32) **Rule S17 (Rules for Tense and Negation)**

If  $\alpha \in P_T$  and  $\delta \in P_{IV}$ , then  $F_{11}(\alpha, \delta), F_{12}(\alpha, \delta), F_{13}(\alpha, \delta), F_{14}(\alpha, \delta), F_{15}(\alpha, \delta) \in P_t$ , where:

- (i)  $F_{11}(\alpha, \delta) = \alpha \delta'$  and  $\delta'$  is the result of replacing the first verb in  $\delta$  by its **negative** third person singular **present**.

<u>Illustration:</u>	(i)	<b>John</b> $\in P_T$ , <b>run</b> $\in P_{IV}$	(Rule S1)
	(ii)	$F_{11}(\text{John}, \text{run}) \in P_t$	(Rule S17)
	(iii)	<b>John doesn't run</b> $\in P_t$	(def. of $F_{11}$ )

- (ii)  $F_{12}(\alpha, \delta) = \alpha \delta'$  and  $\delta'$  is the result of replacing the first verb in  $\delta$  by its third person singular **future**.

<u>Illustration:</u>	(i)	<b>John</b> $\in P_T$ , <b>run</b> $\in P_{IV}$	(Rule S1)
	(ii)	$F_{12}(\text{John}, \text{run}) \in P_t$	(Rule S17)
	(iii)	<b>John will run</b> $\in P_t$	(def. of $F_{12}$ )

- (iii)  $F_{13}(\alpha, \delta) = \alpha \delta'$  and  $\delta'$  is the result of replacing the first verb in  $\delta$  by its **negative** third person singular **future**.

<u>Illustration:</u>	(i)	<b>John</b> $\in P_T$ , <b>run</b> $\in P_{IV}$	(Rule S1)
	(ii)	$F_{13}(\text{John}, \text{run}) \in P_t$	(Rule S17)
	(iii)	<b>John won't run</b> $\in P_t$	(def. of $F_{13}$ )

- (iv)  $F_{14}(\alpha, \delta) = \alpha \delta'$  and  $\delta'$  is the result of replacing the first verb in  $\delta$  by its third person singular **present perfect**.

<u>Illustration:</u>	(i)	<b>John</b> $\in P_T$ , <b>run</b> $\in P_{IV}$	(Rule S1)
	(ii)	$F_{14}(\text{John}, \text{run}) \in P_t$	(Rule S17)
	(iii)	<b>John has run</b> $\in P_t$	(def. of $F_{14}$ )

- (v)  $F_{15}(\alpha, \delta) = \alpha \delta'$  and  $\delta'$  is the result of replacing the first verb in  $\delta$  by its **negative** third person singular **present perfect**.

<u>Illustration:</u>	(i)	<b>John</b> $\in P_T$ , <b>run</b> $\in P_{IV}$	(Rule S1)
	(ii)	$F_{15}(\text{John}, \text{run}) \in P_t$	(Rule S17)
	(iii)	<b>John hasn't run</b> $\in P_t$	(def. of $F_{15}$ )

(33) **Remarks**

- a. Again, Montague doesn't ever properly define the terms *negative...present*, *future*, *negative...future*, *present perfect*, or *negative... present perfect*.
- b. But, again, we could imagine defining an array of morpho-syntactic functions mapping every basic 'verb' to these values.
  - (i)  $\text{Neg3sgPRES}(\text{root}) = \begin{cases} \text{isn't}, & \text{if } \text{root} = \text{be} \\ \dots \\ \text{doesn't root} & \text{otherwise} \end{cases}$
  - (ii)  $3\text{sgFUT}(\text{root}) = \text{will root}$
  - (iii)  $\text{Neg3sgFUT}(\text{root}) = \text{won't root}$
  - (iv)  $3\text{sgPERF}(\text{root}) = \text{has root+ed}$ <sup>3</sup>
  - (v)  $\text{NEG3sgPERF}(\text{root}) = \text{hasn't root+ed}$
- c. Montague doesn't 'decompose' these morphological forms in the way we've come to expect from transformational morpho-syntactic analyses of English.
  - o In PTQ, *negative 3<sup>rd</sup> singular future* and *3<sup>rd</sup> singular future* are just two different primitives; the 'negative' and 'future' isn't separately factored out
- d. Also, these features are introduced by the same rule that adds the subject with the predicate.
  - o Thus, **doesn't run** and **won't run** aren't themselves meaningful expressions.
  - o Thus, we can't get conjunctions like **John doesn't run and won't run**.

(34) **Snarky Side Remark**

Despite how little Montague regards the syntactic research 'emanating from MIT', he could have paid a bit closer attention to *Syntactic Structures* ; )

*Those are all the rules! We can now use them to properly define the set of meaningful expressions for the English fragment.*

(35) **The Phrases of the English Fragment**

$\{ P_A \}_{A \in \text{Cat}}$  is the smallest family of sets indexed by *Cat* such that S1-S17 are true.

(36) **The Meaningful Expressions of the English Fragment**

$\varphi$  is a meaningful expression of the fragment if there is an  $A \in \text{Cat}$  such that  $\varphi \in P_A$ .

---

<sup>3</sup> Actually, what we *really* need here is a separate function PST-PARTICIPLE, mapping every verbal root to its past participle.

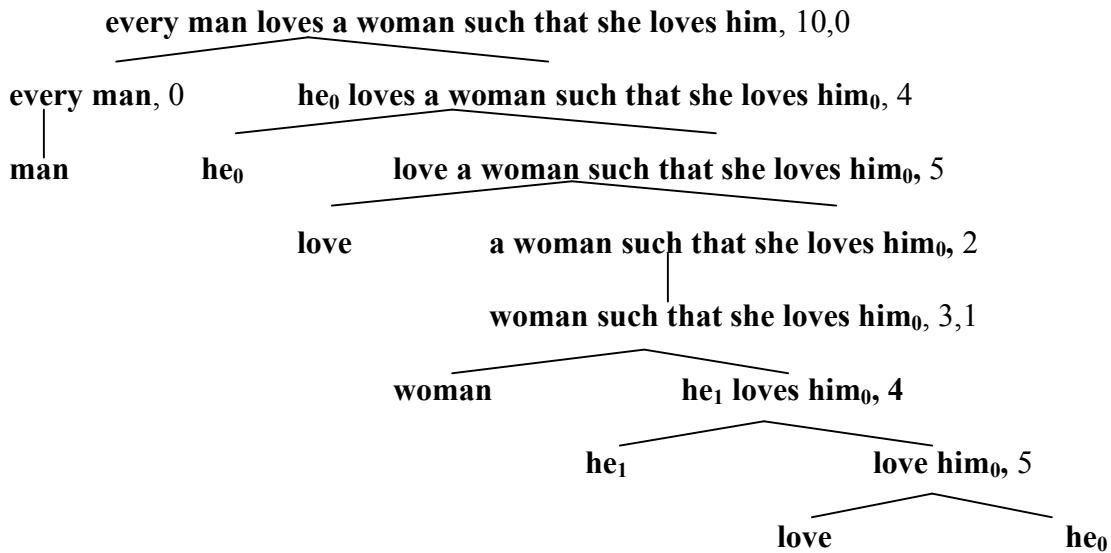
#### 4. Some Illustrations of the Fragment

As the calculations below illustrate, the following are members of  $P_t$

(37) **every man loves a woman such that she loves him**

- (i) **man, woman**  $\in B_{CN}$ , **love**  $\in B_{TV}$ , **he<sub>0</sub>, he<sub>1</sub>**  $\in B_T$  (by (6))
- (ii) **man, woman**  $\in P_{CN}$ , **love**  $\in P_{TV}$ , **he<sub>0</sub>, he<sub>1</sub>**  $\in P_T$  (by S1)
- (iii) **love him<sub>0</sub>**  $\in P_{IV}$  (by S5)
- (iv) **he<sub>1</sub> loves him<sub>0</sub>**  $\in P_t$  (by S4)
- (v) **woman such that she loves him<sub>0</sub>**  $\in P_{CN}$  (by S3)
- (vi) **a woman such that she loves him<sub>0</sub>**  $\in P_T$  (by S2)
- (vii) **love a woman such that she loves him<sub>0</sub>**  $\in P_{IV}$  (by S5)
- (viii) **he<sub>0</sub> loves a woman such that she loves him<sub>0</sub>**  $\in P_t$  (by S4)
- (ix) **every man**  $\in P_T$  (by S2)
- (x) **every man loves a woman such that she loves him**  $\in P_T$  (by S14)

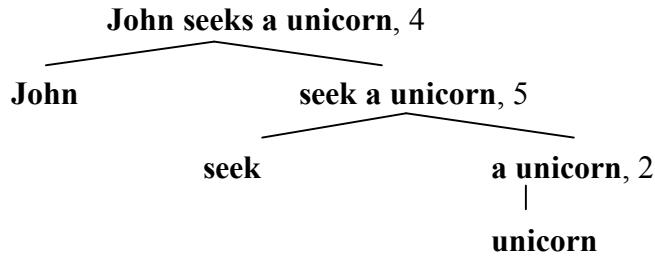
Analysis Tree, Illustrating the Derivation



(38) John seeks a unicorn.

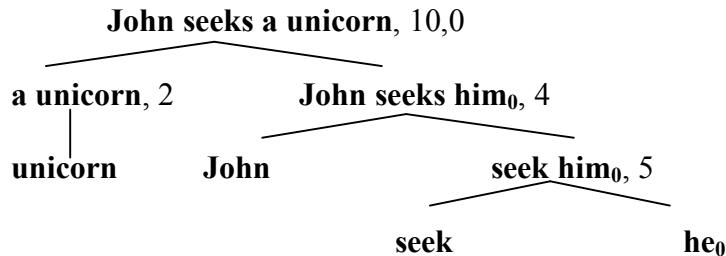
a. Derivation / Analysis One

- (i) **unicorn**  $\in B_{CN}$ , **seek**  $\in B_{TV}$ , **John**  $\in B_T$  (by (6))
- (ii) **unicorn**  $\in P_{CN}$ , **seek**  $\in P_{TV}$ , **John**  $\in P_T$  (by S1)
- (iii) **a unicorn**  $\in P_T$  (by S2)
- (iv) **seek a unicorn**  $\in P_{IV}$  (by S5)
- (v) **John seeks a unicorn**  $\in P_t$  (by S4)



b. Derivation / Analysis Two

- (i) **unicorn**  $\in B_{CN}$ , **seek**  $\in B_{TV}$ , **John**,  $he_0 \in B_T$  (by (6))
- (ii) **unicorn**  $\in P_{CN}$ , **seek**  $\in P_{TV}$ , **John**,  $he_0 \in P_T$  (by S1)
- (iii) **a unicorn**  $\in P_T$  (by S2)
- (iv) **seek him<sub>0</sub>**  $\in P_{IV}$  (by S5)
- (v) **John seeks him<sub>0</sub>**  $\in P_t$  (by S4)
- (vi) **John seeks a unicorn**  $\in P_t$  (by S14)



(39) Remarks

- Again, we see that our English fragment admits of syntactically (and semantically) ambiguous expressions.
- Again, this will present no problems for PTQ, where the translation relation needn't be a *function*.
- In addition, we'll see that under the derivation in (38a), sentence (38) receives the *de dicto* reading, while under the derivation in (38b), it receives *de re* reading

(40) **Some Optional Exercises for Students**

*Show how our English fragment predicts that each of the following are members of  $P_t$*

- a. **John doesn't love the man such that he runs**  
*( $\approx$  John doesn't love the man who runs.)*
- b. **every man will talk about Mary**
- c. **Bill has thought that a man loves him**
- d. **every woman runs or loves Bill**

## The Proper Treatment of Quantification in Ordinary English, Part 2: Intensional Logic

In these notes, we will explore the translation language employed in PTQ, Intensional Logic

### (1) Why Intensional Logic?

- Our English fragment includes such words as **seek**, **conceive**, **believe that**, **wish to**, **allegedly**, **necessarily**, and **about**
  - Notably, these expressions must take the *intensions* of their complements as argument (rather than the *extensions* of their complements)
- Consequently, we will want the induced semantics for the English fragment to end up mapping some English expressions to *their intensions*.
- Thus, since we're developing an indirect interpretation of the fragment, we'll need a logical language that can represent such intensions.
- In addition, this language will have the ability to represent the semantics of the modal elements **necessarily**, *present perfect*, and *future*.

### (2) Some Conceptual Background to Intensional Logic

- In LING 620, you learned that in a sentence like (2a) below, the extension of *thinks* takes as argument the intension of *Mary smokes*.
  - a. John thinks [ that Mary smokes ]
  - b.  $[[ \text{that } \text{XP} ]]^w = [\lambda w' : [[\text{XP}]]^{w'}]$
  - c. (i)  $[[ \text{that } \text{Mary smokes} ]]^w = [\lambda w' : [[\text{Mary smokes}]]^{w'}] = [\lambda w' : \text{Mary smokes in } w']$
- Thus, the **extension** of *that S* is equal to the **intension** of *S*.
- It probably wasn't noted at the time, but observe that *that S* in such a semantics also itself has an intension:
  - d.  $[\lambda w : [[ \text{that } \text{Mary smokes} ]]^w] = [\lambda w : [\lambda w' : \text{Mary smokes in } w']]$
- Thus, the intension of *that S* is a constant function mapping every world to the intension of *S*.

(3) **The Upshot of All This**

The language IL will have an operator (^) that behaves just like *that* in our analysis above

## 1. The Syntax of Intensional Logic (IL)

To define the meaningful expressions of IL, we begin by defining its vocabulary. And, to define its vocabulary, we must first define our system of (intensional) types.

(4) **The System of (Intensional) Types**

The set *Type* is the smallest set such that:

- a.  $e, t \in Type$
- b. If  $\sigma, \tau \in Type$ , then  $\langle\sigma, \tau\rangle \in Type$
- c. If  $\tau \in Type$ , then  $\langle s, \tau \rangle \in Type$

(5) **The Vocabulary of IL**

a. The Logical Constants

(i) *Sentence Connectives*:

$\neg$	Negation	'it is not the case that'
$\wedge$	Conjunction	'and'
$\vee$	Disjunction	'or' (inclusive)
$\rightarrow$	Implication	'if...then'
$\leftrightarrow$	Biconditional	'if and only if'

(ii) *Identity*

$=$	Identity	'equals'
-----	----------	----------

(iii) *Quantifiers*

$\exists$	Existential	'there is an...'
$\forall$	Universal	'for all...'

(iv) *Lambda*

$\lambda$	Lambda	'the function that...'
-----------	--------	------------------------

(v) *Modal Operators*

$\Box$	Necessity	'it is necessary that...'
$\Diamond$	Future	'it will be the case that...'
$\square$	Past	'it has been the case that...'

(vi) *Intension/Extension Ops*

$\wedge$	Up	'the intension of...'
$\vee$	Down	'the extension of...'

The important new additions!

b. The Syntactic Symbols: (,), [, ]

c. The Non-Logical Constants For every type  $\tau \in Type$

(i) An infinite set of constants of type  $\tau$ ,  $Con_\tau$

(ii) A countably infinite set of variables of type  $\tau$ ,  $Var_\tau$

$$Var_\tau = \{v_{n,\tau} : n \in \mathbb{N}\}$$

(6) **Remarks**

- The variable  $v_{n,\tau}$  will sometimes be referred to as ‘the  $n^{\text{th}}$  variable of type  $\tau$ ’
- For some reason, Montague doesn’t enumerate the constants (probably because he doesn’t have to, unlike the variables).
- Montague also never specifies *how* the constants of IL should look.
  - However, he uses  $j, m, b, n$  as meta-language labels for constants of type  $e$
  - Similarly, he uses, **run'**, **man'**, **love'** as meta-language labels for constants of predicative types
- We will follow suit, and temporarily assume the following:
  - $j, m, b, n \in Con_e$
  - **smoke'**, **run'**, **man'**  $\in Con_{<e,t>}$
  - **like'**  $\in Con_{<e,<e,t>}$
  - **think'**  $\in Con_{<<s,t>,<e,t>}$

(7) **The Syntax of Intensional Logic**

- a. Every variable and constant of type  $\tau$  is in  $ME_\tau$
- b. If  $\varphi \in ME_\tau$  and  $u$  is a variable of type  $\sigma$ , then  $\lambda u\varphi \in ME_{<\sigma,\tau>}$
- c. If  $\varphi \in ME_{<\sigma,\tau>}$  and  $\psi \in ME_\sigma$ , then  $\varphi(\psi) \in ME_\tau$
- d. If  $\varphi, \psi \in ME_\tau$ , then  $\varphi = \psi \in ME_t$
- e. If  $\varphi, \psi \in ME_t$  and  $u$  is a variable, then
  - (i)  $\neg\varphi \in ME_t$
  - (ii)  $[\varphi \wedge \psi] \in ME_t$
  - (iii)  $[\varphi \vee \psi] \in ME_t$
  - (iv)  $[\varphi \rightarrow \psi] \in ME_t$
  - (v)  $[\varphi \leftrightarrow \psi] \in ME_t$
  - (vi)  $Vu\varphi \in ME_t$
  - (vii)  $\Lambda u\varphi \in ME_t$
  - (viii)  $\Box\varphi \in ME_t$
  - (ix)  $W\varphi \in ME_t$
  - (x)  $H\varphi \in ME_t$
- f. If  $\varphi \in ME_\tau$ , then  $[^\wedge\varphi] \in ME_{<s,\tau>}$
- g. If  $\varphi \in ME_{<s,\tau>}$ , then  $[^\vee\varphi] \in ME_\tau$

(8) **Meaningful Expression of IL**     $\varphi$  is a meaningful expression of IL if  $\varphi \in \bigcup_{\tau \in Type} ME_\tau$

### (9) Remarks

- a. Note that in (7c) the notation for a predicate applied to its argument is  $\varphi(\psi)$ , and not  $(\varphi \psi)$ , as in TL.
- b. Note that '=' can appear between any two expressions of the same type. Thus, all the following are meaningful expressions:
  - (i)  $j = b$
  - (ii)  $\mathbf{man}' = \mathbf{run}'$
  - (iii)  $\mathbf{man}'(j) = \mathbf{run}'(b)$
 Our semantics for '=' will entail that  $\mathbf{man}'(j) = \mathbf{run}'(b)$  is logically equivalent to  $[\mathbf{man}'(j) \leftrightarrow \mathbf{run}'(b)]$
- c. "The expression  $[\wedge \alpha]$  is regarded as denoting (or as having as its *extension*) the *intension* of the expression  $\alpha$ " (pp. 23-24)
  - o Thus, the operator ' $\wedge$ ' is akin to English *that* in (2b)
- d. "The expression  $[\vee \alpha]$  is meaningful only if  $\alpha$  is an expression that denotes an intension or sense; in such a case  $[\vee \alpha]$  denotes the corresponding extension."
- e. "In the presentation of actual expressions, of intensional logic, square brackets will sometimes for perspicuity be omitted, and sometimes gratuitously inserted."

### (10) Some Illustrative Meaningful Expressions of IL

- a.  $\lambda v_{1,e} [\mathbf{smoke}'(v_{1,e}) \wedge \mathbf{run}'(v_{1,e})]$   
*Informally:* The characteristic function of the set of smokers that run.
- b.  $\lambda v_{1,<e,t>} v_{1,<e,t>}(j)$   
*Informally:* The characteristic function of the set of properties true of John.  
 $(\approx (\lambda P_1 (P_1 \mathbf{john}')))$
- c.  $\lambda v_{1,<e,t>} \forall v_{1,e} [\mathbf{man}'(v_{1,e}) \wedge v_{1,<e,t>}(v_{1,e})]$   
*Informally:* The characteristic function of the set of properties true of a man  
 $(\approx (\lambda P_1 \exists x_1 ((\mathbf{man}' x_1) \& (P_1 x_1))))$
- d.  $\mathbf{think}'([\wedge \mathbf{like}'(j)(m)])(b)$   
*Informally:* 'Bill thinks that Mary likes John'
- e.  $\Box \mathbf{run}'(j)$   
*Informally:* 'It is necessary that John runs'
- f.  $\Lambda v_{1,e} H \mathbf{smoke}'(v_{1,e})$   
*Informally:* 'For all  $x$ , it has been the case that  $x$  smokes.'

## 2. The Semantics of Intensional Logic (IL)

In this section, we will develop a formal model-theoretic semantics for IL. As usual, we begin by defining the notion ‘*denotations of type  $\tau$* ’

- As the following definition suggests, our model structures will be based on a set of entities  $A$ , a set of worlds  $I$ , and a set of times  $J$ .

### (11) The Denotations

Let  $A$ ,  $I$ , and  $J$  be non-empty sets ( $A$  = set of entities;  $I$  = set of possible worlds;  $J$  = set of moments of time). If  $\tau \in Type$ , then the set  $D_{\tau, A, I, J}$  of *possible denotations of type  $\tau$  corresponding to  $A, I, J$*  is defined as follows:

- a.  $D_{e, A, I, J} = A$
- b.  $D_{t, A, I, J} = \{0, 1\}$
- c. If  $\sigma, \tau \in T$  then  $D_{<\sigma, \tau>, A, I, J} =$  the set of functions from  $D_{\sigma, A, I, J}$  to  $D_{\tau, A, I, J}$
- d. If  $\sigma \in T$  then  $D_{<s, \sigma>, A, I, J} =$  the set of functions from  $I \times J$  to  $D_{\sigma, A, I, J}$   
 $= (D_{\sigma, A, I, J})^{I \times J}$

Notes:

- In PTQ, the set  $J$  is the set of times (moments of time); it’s not a set of contexts/variable assignments (unlike in UG).
- In PTQ, a denotation of type  $<s, \sigma>$  is a function from *world-time* pairs to denotations of type  $\sigma$

*In addition to the denotations in (11), our model theoretic semantics for IL will make reference to a set of ‘senses’:*

### (12) The Senses

Let  $A$ ,  $I$ , and  $J$  be non-empty sets ( $A$  = set of entities;  $I$  = set of possible worlds;  $J$  = set of moments of time). If  $\tau \in Type$ , then the set  $S_{\tau, A, I, J}$  of *possible senses of type  $\tau$  corresponding to  $A, I, J$*  is equal to  $D_{<s, \tau>, A, I, J}$

*With these ingredients, we can now define what a model of IL is...*

### (13) A Model of IL

An intensional model (or interpretation) of IL is a quintuple  $<A, I, J, \leq, F>$  such that:

- a.  $A, I, J$  are non-empty sets
- b.  $\leq$  is a linear ordering of  $J$
- c.  $F$  is a function whose domain is the set of constants of IL
- d. Whenever  $\tau \in Type$  and  $\alpha \in Con_{\tau}$ ,  $F(\alpha) \in S_{\tau, A, I, J}$

(14) **Remarks**

Thus, an (intensional) model of IL is defined by: (i) a set of entities A, (ii) a set of worlds I, (iii) a set of times J, (iv) a temporal ordering  $\leq$ , and (v) a ‘lexical’ interpretation function F.

- Note that the function F maps the constants of IL to intensions, **not** denotations.
  - Thus, type e constants are mapped to  $\langle s, e \rangle$  functions (individual concepts)
  - Type  $\langle e, t \rangle$  constants are mapped to  $\langle s, \langle e, t \rangle \rangle$  functions (properties)
  - Type  $\langle e, \langle e, t \rangle \rangle$  constants are mapped to  $\langle s, \langle e, \langle e, t \rangle \rangle \rangle$  functions (‘relations in intension’), etc....
- Models of IL are too complex to specify concretely. Nevertheless, to illustrate key components of the definitions here, we can attempt the following partial description.

(15) **Illustration of an Intensional Model (for IL)**

Let  $\mathcal{M}$  be an intensional model  $\langle A, I, J, \leq, F \rangle$  such that:

- a.  $A = \{\text{Barack, Michele}\}$
- b.  $I = \{w_1, w_2\}$
- c.  $J = \{t_1, t_2, t_3\}$
- d.  $\leq = \{\langle t_1, t_2 \rangle, \langle t_1, t_3 \rangle, \langle t_2, t_3 \rangle, \langle t_1, t_1 \rangle, \langle t_2, t_2 \rangle, \langle t_3, t_3 \rangle\}$
- e. F comprises at least the following mappings:

$$F(b) = \begin{array}{lll} w_1, t_1 \rightarrow & \text{Barack} & ; \\ w_1, t_2 \rightarrow & \text{Barack} & ; \\ w_1, t_3 \rightarrow & \text{Barack} & ; \end{array} \quad \begin{array}{lll} w_2, t_1 \rightarrow & \text{Barack} & \\ w_2, t_2 \rightarrow & \text{Barack} & \\ w_2, t_3 \rightarrow & \text{Barack} & \end{array}$$

$$F(m) = \begin{array}{lll} w_1, t_1 \rightarrow & \text{Michelle} & ; \\ w_1, t_2 \rightarrow & \text{Michelle} & ; \\ w_1, t_3 \rightarrow & \text{Michelle} & ; \end{array} \quad \begin{array}{lll} w_2, t_1 \rightarrow & \text{Michelle} & \\ w_2, t_2 \rightarrow & \text{Michelle} & \\ w_2, t_3 \rightarrow & \text{Michelle} & \end{array}$$

$$F(\text{smoke'}) = \begin{array}{lll} w_1, t_1 \rightarrow & \{\langle \text{Barack}, 1 \rangle, \langle \text{Michelle}, 1 \rangle\} \\ w_1, t_2 \rightarrow & \{\langle \text{Barack}, 1 \rangle, \langle \text{Michelle}, 0 \rangle\} \\ w_1, t_3 \rightarrow & \{\langle \text{Barack}, 0 \rangle, \langle \text{Michelle}, 0 \rangle\} \\ w_2, t_1 \rightarrow & \{\langle \text{Barack}, 0 \rangle, \langle \text{Michelle}, 1 \rangle\} \\ w_2, t_2 \rightarrow & \{\langle \text{Barack}, 0 \rangle, \langle \text{Michelle}, 1 \rangle\} \\ w_2, t_3 \rightarrow & \{\langle \text{Barack}, 0 \rangle, \langle \text{Michelle}, 1 \rangle\} \end{array}$$

F(**think'**) contains the following mapping:

$$w_1, t_1 \rightarrow \left( \begin{array}{l} \left\{ \begin{array}{l} \langle \langle w_1, t_1 \rangle, 1 \rangle \\ \langle \langle w_1, t_2 \rangle, 1 \rangle \\ \langle \langle w_1, t_3 \rangle, 0 \rangle \\ \langle \langle w_2, t_1 \rangle, 0 \rangle \\ \langle \langle w_2, t_2 \rangle, 0 \rangle \\ \langle \langle w_2, t_3 \rangle, 0 \rangle \end{array} \right\} \end{array} \right) \rightarrow \{ \langle \text{Barack}, 1 \rangle, \langle \text{Michelle}, 1 \rangle \}$$

### (16) Remarks Regarding the Model in (15)

- The type-e constant  $b$  ( $m$ ) is interpreted as an  $\langle s, e \rangle$  function (individual concept) that maps every world-time pair to Barack (Michelle).
- The type- $\langle e, t \rangle$  constant **smoke'** is interpreted as an  $\langle s, \langle e, t \rangle \rangle$  function (property). Informally speaking, in this model:
  - At world  $w_1$ , Barack and Michelle both smoke at time  $t_1$ , but then at time  $t_2$ , Michelle quits. Finally, at time  $t_3$ , Barack also quits.
  - At world  $w_2$ , Barack is never a smoker, but Michelle is and never quits.
- The type- $\langle\langle s, t \rangle, \langle e, t \rangle \rangle$  constant **think'** is interpreted as an  $\langle s, \langle\langle s, t \rangle, \langle e, t \rangle \rangle \rangle$  function (the intension of *thinks*). Informally speaking, in this model:
  - At world  $w_1$  and time  $t_1$ , both Barack and Michelle think that Barack smokes.

*Now that we have the definition of a model, the next step is to define the notion of a ‘variable assignment’...*

### (17) Definition of an $\mathcal{M}$ -Assignment

Let  $\mathcal{M}$  be an intensional model  $\langle A, I, J, \leq, F \rangle$ . The function  $g$  is an  $\mathcal{M}$ -assignment if:

- a. It has as its domain the set of all variables of IL.
- b. If  $u$  is a variable of type  $\tau$ , then  $g(u) \in D_{\tau, A, I, J}$

Note: Variable assignments maps variables directly to *denotations (extensions)*, not to *senses (intensions)*.

*With these ingredients, we can now recursively define the central notion ‘extension with respect to an intensional model, world, time, and variable assignment’*

### (18) Extension With Respect to Model, World, Time, and Variable Assignment

Let  $\mathcal{M}$  be an intensional model  $\langle A, I, J, \leq, F \rangle$  for IL, let  $i \in I$  and  $j \in J$ , and let  $g$  be an  $\mathcal{M}$ -assignment. If  $\alpha$  is a meaningful expression of IL, then  $[[\alpha]]^{M,i,j,g}$ , the extension of  $\alpha$  with respect to  $\mathcal{M}$ ,  $i$ ,  $j$ , and  $g$ , is defined as follows:

- a. If  $\alpha$  is a constant, then  $[[\alpha]]^{M,i,j,g} = F(\alpha)(\langle i, j \rangle)$

Note: Thus, we obtain the extension of a constant  $\alpha$  by applying the intension  $F(\alpha)$  to the world-time pair  $\langle i, j \rangle$ .

- b. If  $\alpha$  is a variable, then  $[[\alpha]]^{M,i,j,g} = g(\alpha)$

- c. If  $\alpha \in ME_\tau$  and  $u$  is a variable of type  $\sigma$ , then  $[[\lambda u \alpha]]^{M,i,j,g} =$  the function  $h$  with domain  $D_{\sigma, A, I, J}$  such that whenever  $x$  is in that domain,  $h(x) = [[\alpha]]^{M,i,j,g(u/x)}$
- d. If  $\alpha \in ME_{<\sigma, \tau>}$  and  $\beta \in ME_\sigma$ , then  $[[\alpha(\beta)]]^{M,i,j,g} = [[\alpha]]^{M,i,j,g}([[\beta]]^{M,i,j,g})$
- e. If  $\varphi, \psi \in ME_\tau$ , then  $[[\varphi = \psi]]^{M,i,j,g} = 1$  iff  $[[\varphi]]^{M,i,j,g} = [[\psi]]^{M,i,j,g}$
- f. If  $\varphi, \psi \in ME_t$ , then
  - (i)  $[[\neg \varphi]]^{M,i,j,g} = 1$  iff  $[[\varphi]]^{M,i,j,g} = 0$
  - (ii)  $[[[\varphi \wedge \psi]]]^{M,i,j,g} = 1$  iff  $[[\varphi]]^{M,i,j,g} = 1$  and  $[[\psi]]^{M,i,j,g} = 1$
  - (iii)  $[[[\varphi \vee \psi]]]^{M,i,j,g} = 1$  iff  $[[\varphi]]^{M,i,j,g} = 1$  or  $[[\psi]]^{M,i,j,g} = 1$
  - (iv)  $[[[\varphi \rightarrow \psi]]]^{M,i,j,g} = 1$  iff  $[[\varphi]]^{M,i,j,g} = 0$  or  $[[\psi]]^{M,i,j,g} = 1$
  - (v)  $[[[\varphi \leftrightarrow \psi]]]^{M,i,j,g} = 1$  iff  $[[\varphi]]^{M,i,j,g} = [[\psi]]^{M,i,j,g}$

Note: Thus, if  $\varphi, \psi \in ME_t$ , then  $[[[\varphi \leftrightarrow \psi]]]^{M,i,j,g} = 1$  iff  $[[[\varphi = \psi]]]^{M,i,j,g} = 1$   
Thus, if  $\varphi, \psi \in ME_t$ , then  $[\varphi \leftrightarrow \psi]$  and  $\varphi = \psi$  are logically equivalent.

- g. If  $\varphi \in ME_t$ , and  $u$  is a variable of type  $\tau$ , then
  - (i)  $[[V u \varphi]]^{M,i,j,g} = 1$  iff there is an  $x \in D_{\tau, A, I, J}$  such that  $[[\varphi]]^{M,i,j,g(u/x)} = 1$
  - (ii)  $[[\Lambda u \varphi]]^{M,i,j,g} = 1$  iff for all  $x \in D_{\tau, A, I, J}$ ,  $[[\varphi]]^{M,i,j,g(u/x)} = 1$
- h. If  $\varphi \in ME_t$ , then
  - (i)  $[[\Box \varphi]]^{M,i,j,g} = 1$  iff for all  $i' \in I$  and  $j' \in J$ ,  $[[\varphi]]^{M,i',j',g} = 1$

Note: Thus, ' $\Box \varphi$ ' is more aptly translated as 'necessarily always  $\varphi$ '

  - (ii)  $[[W \varphi]]^{M,i,j,g} = 1$  iff for some  $j' \in J$  such that  $j < j'$ ,  $[[\varphi]]^{M,i,j',g} = 1$
  - (iii)  $[[H \varphi]]^{M,i,j,g} = 1$  iff for some  $j' \in J$  such that  $j' < j$ ,  $[[\varphi]]^{M,i,j',g} = 1$
- i. If  $\alpha \in ME_\tau$ , then  $[[[\wedge \alpha]]]^{M,i,j,g} =$  the function  $h$  with domain  $I \times J$  such that if  $< i', j' > \in I \times J$ , then  $h(< i', j' >) = [[\alpha]]^{M,i',j',g}$

Note: Thus,  $[[[\wedge \alpha]]]^{M,i,j,g}$  is the function mapping a world-time pair to the extension of  $\alpha$  at that world-time. Thus, given (19) below,  $[[[\wedge \alpha]]]^{M,i,j,g}$  is the intension of  $\alpha$  with respect to  $\mathcal{M}$  and  $g$ ,  $[[\alpha]]^{M,g}$

- j. If  $\alpha \in ME_{<s, \tau>}$ , then  $[[[\vee \alpha]]]^{M,i,j,g} = [[\alpha]]^{M,i,j,g}(< i, j >)$

Note: Thus, if the extension of  $\alpha$  at  $i$  and  $j$  is some intension, then the extension of  $[\vee \alpha]$  at  $i$  and  $j$  is that intension applied to  $i$  and  $j$ .

(19) **Intension With Respect to a Model and Variable Assignment**

Let  $\mathcal{M}$  be an intensional model  $\langle A, I, J, \leq, F \rangle$  for IL and let  $g$  be an  $\mathcal{M}$ -assignment. If  $\alpha$  is a meaningful expression of IL, then  $[[\alpha]]^{M,g}$ , the intension of  $\alpha$  with respect to  $\mathcal{M}$  and  $g$ , is the function  $h$  with domain  $I \times J$  such that if  $\langle i', j' \rangle \in I \times J$ , then  $h(\langle i', j' \rangle) = [[\alpha]]^{M, i', j', g}$

(20) **Truth With Respect to a Model, World, and Time**

Let  $\mathcal{M}$  be an intensional model  $\langle A, I, J, \leq, F \rangle$  for IL and let  $i \in I$  and  $j \in J$ . If  $\varphi \in ME_t$ , then  $\varphi$  is true with respect to  $\mathcal{M}$ ,  $i$ , and  $j$  iff for any  $\mathcal{M}$ -assignment  $g$ ,  $[[\varphi]]^{M, i, j, g} = 1$

Let us now illustrate these definitions by using them to interpret meaningful expressions of IL

(21) **Illustrative Computations, Part 1**

In the computations below, let  $\mathcal{M}$  be an intensional model of the kind described in (15).

Let  $g$  be an arbitrary  $\mathcal{M}$ -assignment.

a.  $\text{smoke}'(b)$

$$\begin{array}{lll} (i) & [[\text{smoke}'(b)]]^{M, w_1, t_1, g} & = \quad (\text{by (18d)}) \\ (ii) & [[\text{smoke}']]^{M, w_1, t_1, g}([[b]]^{M, w_1, t_1, g}) & = \quad (\text{by (18a)}) \\ (iii) & F(\text{smoke}')(⟨w_1, t_1⟩)(F(b)(⟨w_1, t_1⟩)) & = \quad (\text{by (15)}) \\ (iv) & \{⟨\text{Barack}, 1⟩, ⟨\text{Michelle}, 1⟩\}(\text{Barack}) & = \\ (v) & 1 & \end{array}$$

b.  $[\wedge \text{smoke}'(b)]$

$$\begin{array}{lll} (i) & [[[\wedge \text{smoke}'(b)]]]^{M, w_1, t_1, g} & = \quad (\text{by (18i)}) \\ (ii) & \text{the function } h \text{ with domain } I \times J \text{ such that} \\ & \text{if } \langle i', j' \rangle \in I \times J, \text{ then } h(\langle i', j' \rangle) = [[\text{smoke}'(b)]]^{M, i', j', g} & = \quad (\text{by (18a,d)}) \\ (iii) & \text{the function } h \text{ with domain } I \times J \text{ such that} \\ & \text{if } \langle i', j' \rangle \in I \times J, \text{ then } h(\langle i', j' \rangle) = F(\text{smoke}')(⟨i', j'⟩)(F(b)(⟨i', j'⟩)) & = \quad (\text{by (15)}) \\ (iv) & \text{the function } h \text{ with domain } I \times J \text{ such that} \\ & \text{if } \langle i', j' \rangle \in I \times J, \text{ then } h(\langle i', j' \rangle) = F(\text{smoke}')(⟨i', j'⟩)(\text{Barack}) & = \quad (\text{by (15)}) \\ (v) & \left\{ \begin{array}{l} \langle \langle w_1, t_1 \rangle, 1 \rangle, \langle \langle w_1, t_2 \rangle, 1 \rangle, \langle \langle w_1, t_3 \rangle, 0 \rangle, \\ \langle \langle w_2, t_1 \rangle, 0 \rangle, \langle \langle w_2, t_2 \rangle, 0 \rangle, \langle \langle w_2, t_3 \rangle, 0 \rangle \end{array} \right\} & \end{array}$$

(22) **Illustrative Computations, Part 2**

In the computations below, let  $\mathcal{M}$  be an intensional model of the kind described in (15). Let  $g$  be an arbitrary  $\mathcal{M}$ -assignment.

a.  $\square b = b$

- (i)  $[[\square b = b]]^{M,w1,t1,g} = 1 \quad \text{iff} \quad (\text{by (18h)})$
- (ii) for all  $i' \in I$  and  $j' \in J$ ,  $[[b = b]]^{M,i',j',g} = 1 \quad \text{iff} \quad (\text{by (18e)})$
- (iii) for all  $i' \in I$  and  $j' \in J$ ,  $[[b]]^{M,i',j',g} = [[b]]^{M,i',j',g} \quad \text{iff} \quad (\text{by (18a)})$
- (iv) for all  $i' \in I$  and  $j' \in J$ ,  $F(b)(\langle i', j' \rangle) = F(b)(\langle i', j' \rangle)$
- (v) Thus,  $[[\square b = b]]^{M,w1,t1,g} = 1$

b.  $H \text{ smoke}'(b)$

- (i)  $[[H \text{ smoke}'(b)]]^{M,w1,t2,g} = 1 \quad \text{iff} \quad (\text{by (18h)})$
- (ii) for some  $j' \in J$  such that  $j' < t_2$ ,  $[[\text{smoke}'(b)]]^{M,w1,j',g} = 1 \quad \text{iff} \quad (\text{by (18a,d)})$
- (iii) for some  $j' \in J$  such that  $j' < t_2$ ,  $F(\text{smoke}')(\langle w_1, j' \rangle)(F(b)(\langle w_1, j' \rangle)) = 1 \quad \text{iff} \quad (\text{by (15)})$
- (iv) for some  $j' \in J$  such that  $j' < t_2$ ,  $F(\text{smoke}')(\langle w_1, j' \rangle)(\text{Barack}) = 1$
- (v) Given that  $F(\text{smoke}')(\langle w_1, t_1 \rangle)(\text{Barack}) = 1$ ,  $[[H \text{ smoke}'(b)]]^{M,w1,t2,g} = 1$

c.  $W \text{ smoke}'(b)$

- (i)  $[[W \text{ smoke}'(b)]]^{M,w1,t2,g} = 1 \quad \text{iff} \quad (\text{by (18h)})$
- (ii) for some  $j' \in J$  such that  $t_2 < j'$ ,  $[[\text{smoke}'(b)]]^{M,w1,j',g} = 1 \quad \text{iff} \quad (\text{by (18a,d)})$
- (iii) for some  $j' \in J$  such that  $t_2 < j'$ ,  $F(\text{smoke}')(\langle w_1, j' \rangle)(F(b)(\langle w_1, j' \rangle)) = 1 \quad \text{iff} \quad (\text{by (15)})$
- (iv) for some  $j' \in J$  such that  $t_2 < j'$ ,  $F(\text{smoke}')(\langle w_1, j' \rangle)(\text{Barack}) = 1$
- (v) Given that  $F(\text{smoke}')(\langle w_1, t_3 \rangle)(\text{Barack}) = 0$ ,  $[[W \text{ smoke}'(b)]]^{M,w1,t2,g} = 0$

d.  $\text{think}'([\wedge \text{smoke}'(b)])(m) \quad (\text{'Michelle thinks that Barack smokes'})$

- (i)  $[[\text{think}'([\wedge \text{smoke}'(b)])(m)]]^{M,w1,t1,g} = \quad (\text{by (18d)})$
- (ii)  $[[\text{think}']]^{M,w1,t1,g}([[[\wedge \text{smoke}'(b)]]]^{M,w1,t1,g})([[m]]^{M,w1,t1,g}) = \quad (\text{by (18a)})$
- (iii)  $F(\text{think}')(\langle w_1, t_1 \rangle)([[[\wedge \text{smoke}'(b)]]]^{M,w1,t1,g})(F(m)(\langle w_1, t_1 \rangle)) = \quad (\text{by (15)})$
- (iv)  $F(\text{think}')(\langle w_1, t_1 \rangle)([[[\wedge \text{smoke}'(b)]]]^{M,w1,t1,g})(\text{Michelle}) = \quad (\text{by (21b)})$
- (v)  $F(\text{think}')[\langle w_1, t_1 \rangle]$   
 $(\{\langle w_1, t_1 \rangle, 1, \langle w_1, t_2 \rangle, 1, \langle w_1, t_3 \rangle, 0,$   
 $\langle w_2, t_1 \rangle, 0, \langle w_2, t_2 \rangle, 0, \langle w_2, t_3 \rangle, 0\})(\text{Michelle}) = \quad (\text{by (15)})$
- (vi) 1

(23) **Optional Exercise for the Student**

Compute the value of  $\lambda v_{1,<e,t>} V v_{1,e} [\text{smoke}'(v_{1,e}) \wedge v_{1,<e,t>}(v_{1,e})]$  ]<sup>M,w1,t1,g</sup>

Show that it is the characteristic function of  $\langle et \rangle$ -functions that are true of some smoker.

### 3. Some Key Validities of Intensional Logic

In this section, we'll make note of some key validities in IL, which we will use to 'convert' the translations of English expressions to simpler, logically equivalent formulae.

## (24) Alpha Conversion

If variable  $v$  is bound in  $\varphi$ , and variable  $v'$  appears nowhere in  $\varphi$ , then  $\varphi$  is logically equivalent to  $[v/v']\varphi$

*Although alpha-conversion is just the same as before, lambda-conversion is now subject to an important new restriction.*

## (25) Lambda Conversion

Let  $(\lambda v\psi)$  and  $\varphi$  be meaningful expressions with no variables in common, and let  $\varphi \in \text{ME}_\tau$  and  $v \in \text{Var}_\tau$ . If (a) or (b) hold, then the expressions in (c) are logically equivalent.



### (26) Why This New Restriction on Lambda Conversion?

- Consider, for example, the formulae  $[\lambda v W\chi](\varphi)$  and  $[\varphi/v]W\chi$
  - Consider their extensions at a given world and time:  
 $[[ [\lambda v W\chi](\varphi) ]]^{M,i,j,g}$  and  $[[ [\varphi/v]W\chi ]]^{M,i,j,g}$
  - When we compute  $[[ [\lambda v W\chi](\varphi) ]]^{M,i,j,g}$ , we will compute  $[[ [\lambda v W\chi] ]]^{M,i,j,g}$  and apply it to the value  $[[\varphi]]^{M,i,j,g}$
  - However, when we compute  $[[ [\varphi/v]W\chi ]]^{M,i,j,g}$ , we *don't* compute  $[[\varphi]]^{M,i,j,g}$ . Rather, we will end up computing  $[[\varphi]]^{M,i,j',g}$  for some  $j' > j$ .
  - Therefore, **unless  $\varphi$  has the same extension for every world and time**, we can't guarantee that  $[[ [\lambda v W\chi](\varphi) ]]^{M,i,j,g}$  will be the same value as  $[[ [\varphi/v]W\chi ]]^{M,i,j,g}$

(27) An Analogy to Maybe Make (25)-(26) More Intuitive

- In LING 620, we learned that (a) is ambiguous, and has both the readings in (b).
  - a. John thinks that the president smokes.
  - b.
    - (i) *De Dicto Reading*:  
In all of John's 'belief worlds'  $w$ ', the president in  $w'$  smokes in  $w'$ .
    - (ii) *De Re Reading*:  
In all of John's 'believe worlds'  $w$ ', the president in  $w_0$  smokes in  $w'$ .
- We also saw that we can get reading (i) from an LF where the phrase *the president* is in the scope of 'believes', while we get (ii) from an LF where it's moved above the scope of *believes*.
  - c.
    - (i) *De Dicto LF*: [ John [ believes [ **the president** smokes ] ] ]
    - (ii) *De Re LF*: [ **the president** [ 1 [ John [ believes [  $t_1$  smokes ] ... ] ] ]
- This is because in LF (i), *the president* ends up semantically evaluated with respect to the belief worlds, while in LF (ii), it is evaluated with respect to the actual world.
- Consequently, we saw that for proper names, the two LFs in (c) end up mapped to the same interpretation, **because proper names have the same value in every world**
- d. Two LFs Mapped to the Same Truth-Conditions
  - (i) [ John [ believes [ Mary smokes ] ] ]
  - (ii) [ Mary [ 1 [ John [ believes [  $t_1$  smokes ] ... ] ] ]
- The same general issue here is also at play for IL in (25)-(26).
  - The IL operators  $\wedge$ ,  $\Box$ , H, W *shift* the evaluation worlds and time.
  - Thus, if  $\varphi$  is in the scope of  $\wedge$ ,  $\Box$ , H, W, we *won't* interpret  $\varphi$  with respect to the 'actual' world/time
  - Consequently, if the variable  $v$  in  $\psi$  is in the scope of  $\wedge$ ,  $\Box$ , W, F, then in (i)  $\varphi$  is interpreted with respect to the initial, 'matrix' evaluation world/time, while in (ii), it is not.
    - (i)  $[\lambda v \psi](\varphi)$
    - (ii)  $[\varphi/v]\psi$
  - However, **if  $\varphi$  has the same extension in every world and time, then this difference won't matter for the resulting interpretation, and (i) and (ii) will be logically equivalent.**

(28) **One Last Important Note**

- Given the definition in (18b), it follows that if  $\varphi$  is a variable, then for all  $i, i' \in I$  and  $j, j' \in J$ ,  $[[\varphi]]^{M,i,j,g} = [[\varphi]]^{M,i',j',g} = g(\varphi)$
- Therefore, if  $\varphi$  is a variable, then as long as  $[\lambda v \psi](\varphi)$  and  $\varphi$  have no variables in common, then the following *will always* be logically equivalent (even in IL):
 

(i) $[\lambda v \psi](\varphi)$	(ii) $[\varphi/v]\psi$
---------------------------------	------------------------

*In addition to alpha-conversion and lambda-conversion, our translations in PTQ will also make use of the key logical equivalence in (29)...*

(29) **Down-Up Cancellation (Dowty et al. 1981)**

The following two expressions are logically equivalent:

- $\alpha$
- $[^\vee [^\wedge \alpha]]$

- That is, put informally, the extension of  $\alpha$  at world  $i$  and time  $j$  will always be equal to the intension of  $\alpha$  applied to the world  $i$  and the time  $j$ .

(30) **Optional Exercise for Students**

Let  $\mathcal{M}$  be an intensional model  $\langle A, I, J, \leq, F \rangle$ ,  $i \in I$ ,  $j \in J$ , and  $g$  be an arbitrary  $\mathcal{M}$ -assignment. Show that  $[[[\alpha]]^{M,i,j,g}] = [[ [^\vee [^\wedge \alpha]] ]]^{M,i,j,g}$

#### 4. Some Useful Abbreviations in PTQ

*In PTQ, Montague makes use of a large number of meta-linguistic abbreviatory conventions.*

- |                                       |                           |                    |                                 |
|---------------------------------------|---------------------------|--------------------|---------------------------------|
| (31) <b>Relational Notation:</b>      | $\gamma(\alpha, \beta)$   | <u>abbreviates</u> | $\gamma(\alpha)(\beta)$         |
| (32) <b>Curly Bracket Notation 1:</b> | $\gamma\{\alpha\}$        | <u>abbreviates</u> | $[^\vee \gamma](\alpha)$        |
| (33) <b>Curly Bracket Notation 2:</b> | $\gamma\{\alpha, \beta\}$ | <u>abbreviates</u> | $[^\vee \gamma](\alpha, \beta)$ |

- (34) **Key Illustration:** Let  $P$  be a variable of type  $\langle s, \langle e, t \rangle \rangle$

$[\lambda P P\{b\}](^{\text{smoke}}')$	$\Leftrightarrow$	(by $\lambda$ -conversion)
$[^{\text{smoke}}']\{b\}$	$=$	(by Curly Bracket Notation (CBN))
$[^\vee [^\wedge \text{smoke}'](b)$	$\Leftrightarrow$	(by Down-Up Cancellation (DUC))
$\text{smoke}'(b)$		

(34) **The Curvy Hat**

- Montague puts a curvy ‘hat’ over a variable to indicate lambda abstraction over that variable.
- I will follow Dowty *et al.* 1981 in not making use of this notation
  - After all, ‘ $\lambda$ ’ is clear and easy enough...

(35) **The Sharp Hat**       $\hat{u}\varphi$       abbreviates       $[\wedge\lambda u\varphi]$

- I will not make much use of this abbreviation.
- However, do familiarize yourself with it, as Montague uses it *a lot*.

## The Proper Treatment of Quantification in Ordinary English, Part 3: The Translation System, Part 1<sup>1</sup>

In these notes, we will begin to explore Montague's system for mapping expressions in our fragment of English to formulae of Intensional Logic.

### 1. The Category-to-Type Mapping

The first key ingredient of a PTQ-style translation system (and a UG-style translation base) is a function mapping the categories of the natural language to the categories of the logical language (*i.e.*, the types).

- *It is at this first step that the semantic analysis developed in PTQ immediately gets rather complicated.*
- *Understanding this mapping and why it is this way is key to understanding everything that follows...*

#### (1) The Core Issue: Opaque Environments in English

- Recall from the handout “*The Fragment of English*” that our fragment of English contains the following basic expressions: **seek**, **allegedly**, **about**
- Recall also that these expressions create an ‘opaque’ (non-extensional) environment.
  - “John seeks a unicorn” doesn’t entail “there is a unicorn.”
  - “John talked about a unicorn” doesn’t entail “there is a unicorn.”
  - “John allegedly danced” doesn’t entail “John danced.”
- Consequently, the meanings (extensions) of these lexical items ([[X]]) must combine with the intension of their complements
  - [[**seek**]] takes the intension of **a unicorn** as argument
  - [[**about**]] takes the intension of **a unicorn** as argument
  - [[**allegedly**]] takes the intension of **dance** as argument.
- Thus, [[**seek**]] is not of type  $\langle e, \langle e, t \rangle \rangle$ , and so in our indirect semantics, the translation of **seek** cannot be of type  $\langle e, \langle e, t \rangle \rangle$ 
  - Similarly, the translation of **about** cannot be of type  $\langle e, \langle e, t \rangle \rangle$
  - Similarly, the translation of **allegedly** cannot be of type  $\langle\langle e, t \rangle, \langle e, t \rangle \rangle$

#### (2) Burning Question: What should the types of [[**seek**]], [[**about**]], [[**allegedly**]] be?

Before we develop an answer to this question in (2), let us notice two additional important issues

<sup>1</sup> These notes are based upon material in Dowty *et al.* (1981) Chapter 7.

### (3) Issue 1: Higher, Intensional Types for *Everything*

- Since **seek** is a TV, and its translation is *not* of type  $\langle e, \langle e, t \rangle \rangle$ , then – due to the need for category-to-type correspondence in the translation – **no TV can have a translation of type  $\langle e, \langle e, t \rangle \rangle$** 
  - Therefore, even the translations of **find**, **eat**, and **love** will be of the same high, intensional type as that of **seek**
- Similarly, since **allegedly** is an IAV, it follows that **no IAV (not even ‘rapidly’) can have a translation of type  $\langle\langle e, t \rangle, \langle e, t \rangle \rangle$** 
  - Therefore, the translation of **rapidly** must be of the same, high, intensional type as **allegedly**
- Similarly, since **about** is an IAV/T (preposition), it follows that **no IAV/T (not even ‘in’) can have a translation of type  $\langle e, \langle e, t \rangle \rangle$** 
  - Therefore, the translation of **in** must also be of the same, high, intnsional type as **about**

### (4) Immediate Problem: Failure to Predict Key Inferences

If the translations (and induced interpretations) of **eat**, **rapidly**, and **in** are of the same high, intensional types as **seek**, **allegedly**, and **about**, our semantics will fail to predict the validity of the following inferences:

- a. “John ate a unicorn” *does* entail “There is a unicorn.”
- b. “John talked in a house” *does* entail “There is a house.”
- c. “John rapidly danced” *does* entail “John danced.”

### (5) The Solution (Preview)

As we’ll see later, we can solve the problem in (4) by adding ‘meaning postulates’ for **eat**, **rapidly**, **in** that guarantee the inferences in (4).

- Thus, by adding these meaning postulates, we guarantee that these lexical items behave *as if* they received a purely extensional semantics (even though they don’t).

(6) **Issue 2: All ‘Function Application’ is *Intensional* Function Application**

- Given (3), all TVs will have an (induced) meaning that takes the *intension* of a term as its argument:
  - Both **seek** and **eat** take the *intension* of **a unicorn** as argument.
- Given (3), all IAVs will have an (induced) meaning that takes the *intension* of an IV as its argument.
  - Both **allegedly** and **rapidly** take the *intension* of **dance** as argument.
- Given (3), all IAV/Ts will have an (induced) meaning that takes the *intension* of a term as its argument.
  - Both **about** and **in** take the *intension* of **a unicorn** as argument.
- Also, note that the (induced) meaning of IV/t verbs like **believe that** take as argument the *intension* of their sentential complements.
- Similarly, the (induced) meaning of IV//IV verbs like **try to** take as argument the *intension* of their IV complements

**The General Pattern Emerging Here:**

The translation (induced meaning) of a predicative expression in English always takes as argument the *intension* of the translation (induced meaning) of its syntactic argument.

(7) **Immediate Consequence: Higher Types for the Translations of Terms**

- Recall that we view Ts (= t/IV) as being predicates of IVs
- Thus, the general pattern in (6) entails that, in sentence (7a) below, the translation of a T like **John** should take as argument the *intension* of the translation of **smokes**
  - John smokes.**
- Thus, the translation of **John** should be something like  $[\lambda v_{0,<s,<e,t>>} v_{0,<s,<e,t>>} \{j\}]$

Rough Illustration:

- |       |   |                   |                               |
|-------|---|-------------------|-------------------------------|
| (i)   | The translation of <b>John smokes</b>   | =                 | (by (6))                      |
| (ii)  | The translation of <b>John</b> taking as argument the intension of the translation of <b>smokes</b> . | =                 | (by assumption, and notation) |
| (iii) | $[\lambda v_{0,<s,<e,t>>} v_{0,<s,<e,t>>} \{j\}]([\wedge \text{smokes}'])$                            | $\Leftrightarrow$ | (by $\lambda$ -conversion)    |
| (iv)  | $[\wedge \text{smokes}']\{j\}$  | =                 | (by Curly Bracket Notation)   |
| (v)   | $[\vee [\wedge \text{smokes}']]\{j\}$   | $\Leftrightarrow$ | (by Down-Up Cancellation)     |
| (vi)  | $\text{smokes}'(j)$   |                   |                               |

All these observations taken together lead us (and Montague) to the following general view of the category-to-type correspondence in PTQ...

(8) **Montague's Category-to-Type Correspondence in PTQ**

The function  $f$  has *Cat* as its domain and is such that:

- a.  $f(e) = e$
- b.  $f(t) = t$
- c.  $f(A/B) = f(A//B) = \langle \langle s, f(B) \rangle, f(A) \rangle$

(9) **Remarks**

Thus, a predicative expression in English (*i.e.*, one of category A/B or A//B) will always get a translation that takes as argument an *intensional* expression (type  $\langle s, f(B) \rangle$ )

- Thus, if  $\varphi$  in English combines syntactically with an expression of category B (*i.e.*,  $\varphi$  is of category A/B)...
  - Then the translation of  $\varphi$  will combine syntactically with the *intension* of the translation of an expression of category B (*i.e.*,  $\langle s, f(B) \rangle$ )
- Thus, we immediately capture the higher intensional types of the translations of IVs, IAVs, IAV/Ts, IV/ts, and IV//IVs

(10) **A Possible Snag**

As beautiful as (8) is, it entails the following category-to-type correspondences.

- a.  $f(CN) = f(t//e) = \langle \langle s, e \rangle, t \rangle$
- b.  $f(IV) = f(t/e) = \langle \langle s, e \rangle, t \rangle$
- c.  $f(T) = f(t/(t/e)) = \langle \langle s, \langle \langle s, e \rangle, t \rangle \rangle, t \rangle$

- Thus, (8) predicts that the translation of **walk** will be a function of type  $\langle \langle s, e \rangle, t \rangle$ 
  - Thus, the translation (meaning) of **walk** takes *individual concepts* as argument
- Consequently, terms end up translated to expressions of *extremely* high type

(11) **But, Is This Really a ‘Snag’**

- a. A ‘Pro’ of the Correspondences in (10):

By treating all IVs as (indirectly) denoting  $\langle \langle s, e \rangle, t \rangle$  functions, Montague can analyze such puzzling sentences as “The temperature is 90 and rising.”

- b. ‘Cons’ of the Correspondences in (10):

- As detailed by Dowty *et al.* (1981), there may be other, superior analyses of “The temperature is 90 and rising.”
- The resulting translations for (relatively simple) expressions of English get rather complicated.

(12) **A Simpler Category-to-Type Correspondence**

As discovered by Bennett (1976) and detailed by Dowty *et al.* (1981), the following category-to-type mapping necessitates very few changes to the overall PTQ system.

The function  $f$  has *Cat* as its domain and is such that:

- a.  $f(e) = e$
- b.  $f(t) = t$
- c.  $f(CN) = f(IV) = \langle e, t \rangle$
- d. For all other categories  $A/B$ ,  $f(A/B) = f(A//B) = \langle \langle s, f(B) \rangle, f(A) \rangle$

(13) **Category-to-Type Correspondences, Under (12)**

Category	Corresponding Type	Set-Theoretic Object
$t$ (sentences)	$t$	truth-value
$CN$	$\langle e, t \rangle$	function from entities to truth-values
$IV$	$\langle e, t \rangle$	function from entities to truth-values
$T (= t/IV)$	$\langle \langle s, \langle e, t \rangle \rangle, t \rangle$	function from properties to truth-values <sup>2</sup>
$IV/t$	$\langle \langle s, t \rangle, \langle e, t \rangle \rangle$	function from propositions to $\langle e, t \rangle$ -functions
$IV//IV$	$\langle \langle s, \langle e, t \rangle \rangle, \langle e, t \rangle \rangle$	function from properties to $\langle e, t \rangle$ -functions
$IAV (= IV/IV)$	$\langle \langle s, \langle e, t \rangle \rangle, \langle e, t \rangle \rangle$	function from properties to $\langle e, t \rangle$ -functions
$TV (= IV/T)$	$\langle \langle s, \langle \langle s, \langle e, t \rangle \rangle, t \rangle \rangle, \langle e, t \rangle \rangle$	<b>function from the intensions of GQs to an <math>\langle e, t \rangle</math>-function.</b>

I will follow Bennett (1976) and Dowty *et al.* (1981) in adopting the category-to-type correspondences in (12)-(13).

- Again, these will not necessitate any serious changes from what's in the original article...
- The benefit is that the system ends up being relatively simpler...

## 2. The ‘Lexical’ Translation Function

The second main ingredient of a PTQ-style translation system (and the third main ingredient of a UG-style translation base) is a function translating the basic expressions of the natural language.

- Again, in PTQ, the range of this function can only be *constants* of the logical language.
- Consequently, this function in PTQ will *not* have the following basic expressions in its range (since we want them translated as either (i) variables, or (ii) complex expressions)
  - $B_T$  The basic terms of English, {John, Mary, Bill, ninety, he<sub>0</sub>, he<sub>1</sub>, he<sub>2</sub>, ...}
  - **necessarily**
  - **be**

<sup>2</sup> Such functions, I will (somewhat misleadingly) refer to as ‘GQs’

(14) **The ‘Lexical’ Translation Function**

Let  $g$  be a function such that:

- a. The domain of  $g$  is the set of basic expressions of our fragment of English other than **be**, **necessarily**, and the members of  $B_T$
- b. Whenever  $A \in Cat$ ,  $\alpha \in B_A$ , and  $\alpha$  is in the domain of  $g$ ,  $g(\alpha) \in Con_{f(A)}$

(15) **A Picture of the ‘Lexical’ Translation Function**

- a. The IVs:  $g(\mathbf{run}), g(\mathbf{walk}), g(\mathbf{talk}), g(\mathbf{rise}), g(\mathbf{change}) \in Con_{<e,t>}$
- b. The CNs:  $g(\mathbf{man}), g(\mathbf{woman}), g(\mathbf{park}), g(\mathbf{fish}), g(\mathbf{pen}), g(\mathbf{unicorn}), g(\mathbf{price}), g(\mathbf{temperature}) \in Con_{<e,t>}$
- c. The IV/t's:  $g(\mathbf{believe\ that}), g(\mathbf{assert\ that}) \in Con_{<<s,t>,<e,t>}$
- d. The IV//IVs:  $g(\mathbf{try\ to}), g(\mathbf{wish\ to}) \in Con_{<<s,<e,t>>,<e,t>}$
- e. The IAVs:  $g(\mathbf{rapidly}), g(\mathbf{slowly}), g(\mathbf{voluntarily}), g(\mathbf{allegedly}) \in Con_{<<s,<e,t>>,<e,t>}$
- f. The TVs:  $g(\mathbf{find}), g(\mathbf{lose}), g(\mathbf{eat}), g(\mathbf{love}), g(\mathbf{date}), g(\mathbf{seek}), g(\mathbf{conceive}) \in Con_{<<s,<<s,<e,t>>,t>>,<e,t>>}$
- g. The IAV/Ts:  $g(\mathbf{in}), g(\mathbf{about}) \in Con_{<<s,<<s,<e,t>>,t>>,<<s,<e,t>>,<e,t>>>}$

(16) **Some Useful Meta-Linguistic Abbreviations**

- Recall that Montague never specifies *what* the constants of IL look like.
- Nevertheless, it will be very useful to have some elegant, compact way of referring to the translations of various basic expressions of English.
- Montague therefore introduces the following convention for forming *meta-linguistic* labels / abbreviations for various translations

- a. The Convention: If  $\alpha$  is in the domain of  $g$ , then  $\alpha' = g(\alpha)$
- b. Illustration:
  - (i)  $\mathbf{run}' = g(\mathbf{run})$  [i.e., whatever  $Con_{<e,t>}$   $g$  maps **run** to]
  - (ii)  $\mathbf{man}' = g(\mathbf{man})$  [i.e., whatever  $Con_{<e,t>}$   $g$  maps **man** to]
  - (iii)  $\mathbf{find}' = g(\mathbf{find})$  [i.e., whatever  $Con_{<<s,<<s,<e,t>>,t>>,<e,t>>}$   $g$  maps **find** to]
  - (iv)  $\mathbf{believe\ that}' = g(\mathbf{believe\ that})$  [i.e., whatever  $Con_{<<s,t>,<e,t>>}$   $g$  maps **believe that** to]

## (17) Some Further Useful Meta-Linguistic Abbreviations

- a. In our translation rule for proper names, we will want to make reference to certain specified members of  $\text{Cone}$ . Montague introduces  $j$ ,  $m$ ,  $b$ ,  $n$  as meta-linguistic labels for such constants.

  - Again, the letters ‘ $j$ ’, ‘ $m$ ’, ‘ $b$ ’, and ‘ $n$ ’ aren’t (necessarily) constants.
  - Rather,  $j$ ,  $m$ ,  $b$ ,  $n$  are simply labels we are using to refer to such constants.

b. In our translation rules, we will also want to use certain specific variables. Therefore, to save space, Montague introduces the following abbreviations:

(i)	$x, y, x_n$	<i>abbreviates</i>	$v_{1,e}, v_{3,e} v_{2n,e}$	(entity variable)
(ii)	$p$	<i>abbreviates</i>	$v_{0,<s,t>}$	(proposition variable)
(iii)	$P, Q$	<i>abbreviate</i>	$v_{0,<s,<e,t>>} , v_{1,<s,<e,t>>}$	(property variable)
(iv)	$\mathcal{P}$	<i>abbreviates</i>	$v_{0,<s,<<s,<e,t>>,t>>}$	(variable over intensions of GQs)

*With all these ingredients in place, we are now ready to lay out the translation rules of PTO!*

### **3. Some Translation Rules of PTQ**

We will begin with the first translation rule, which governs the basic expressions of the fragment

(18) **Rule T1 (For Basic Expressions)**

- a. If  $\alpha$  is in the domain of  $g$ , then  $\alpha$  translates into  $g(\alpha)$

<i>Illustration:</i>	<b>run</b>	<u>translates to</u>	<b>run'</b> (= $g(\text{run})$ )
	<b>man</b>	<u>translates to</u>	<b>man'</b> (= $g(\text{man})$ )
	<b>find</b>	<u>translates to</u>	<b>find'</b> (= $g(\text{find})$ )
	<b>believes that</b>	<u>translates to</u>	<b>believes that'</b> (= $g(\text{believes that})$ )

- b.      **be**      translates into  $\lambda P \lambda x. P\{[^{\wedge} y] [^v x = ^v y]\}$

Note: Don't worry about this for now; we may discuss it later.

- c. necessarily translates into  $\lambda p \ [ \Box^v p ]$

Note: Again, don't worry about this one for now.

- d.
- (i) **John** translates into  $\lambda P[P\{j\}]$
- (ii) **Mary** translates into  $\lambda P[P\{m\}]$
- (iii) **Bill** translates into  $\lambda P[P\{b\}]$
- (iv) **ninety** translates into  $\lambda P[P\{n\}]$

Note:

- In PTQ itself, the actual translation of (e.g.) **John** is  $\lambda P[P\{\cdot j\}]$
- This is because in PTQ, Montague held IVs like **walk** to be of type  $\langle\langle s, e \rangle, t \rangle$
- Since we're assuming the simpler category-to-type mapping in (12), we can use the simpler translations in (18d) above.

- e. **he<sub>n</sub>** translates into  $\lambda P[P\{x_n\}]$

Thus, translation rule T1 will cover the translations of all the basic expressions in our fragment.

- From this point on, our discussion of the translation rules will not follow the order of the rules in PTQ.
- Rather, as with Dowty *et al.* (1981), we will discuss the rules in a more ‘pedagogically oriented’ order.
- We’ll thus next turn directly for the translation rule handling subject-predicate structures.

### (19) Rule T4 (Subject-Predicate Translation Rule)

If  $\delta \in P_{t/IV}$ ,  $\beta \in P_{IV}$ , and  $\delta$ ,  $\beta$  translate into  $\delta'$ ,  $\beta'$  respectively, then  $F_4(\delta, \beta)$  translates into  $\delta'([\wedge\beta'])$

Note: Just as promised in (7), the translation of a term  $\delta$  will take as argument the *intension* of the translation of an IV  $\beta$ .

### (20) Illustration of Rule T4

- a. Obtaining the Translation
  - (i) **John** translates into  $\lambda P[P\{j\}]$ , **run** translates into **run'** (T1)
  - (ii)  $F_4(\text{John}, \text{run})$  translates into  $[\lambda P[P\{j\}]](\wedge\text{run}')$  (T4)
  - (iii) **John runs** translates into  $[\lambda P[P\{j\}]](\wedge\text{run}')$  (def. of  $F_4$ )
- b. Simplifying the Translation
  - (i)  $[\lambda P[P\{j\}]](\wedge\text{run}')$   $\Leftrightarrow$  ( $\lambda$ -Conv.)
  - (ii)  $[\wedge\text{run}']\{j\}$   $\Leftrightarrow$  (CBN)
  - (iii)  $[\vee[\wedge\text{run}']]\{j\}$   $\Leftrightarrow$  (DUC)
  - (iv) **run'(j)**

(21) **Remarks**

- Thus, the translation of **John runs** is logically equivalent to **run'(j)**
- Thus, under our induced semantics for English, we predict that relative to a model  $\mathcal{M}$ , a world i, a time j, and a variable assignment g, **John runs** will be 1 iff
  - The extension of **run'** w.r.t.  $\mathcal{M}, i, j, g$  maps the extension of *j* w.r.t.  $\mathcal{M}, i, j, g$  to 1
  - (Informally speaking) ‘John is running’ in world i at time j (in model  $\mathcal{M}$ )
- **Thus, we find that our induced semantics correctly predicts the truth-conditions of present tense sentences.**
  - Their truth at a particular world/time depends upon the extensions of the predicates at that world/time

(22) **An Important Addition: Meaning Postulates for Translations of Names**

- Note that, in order to *really* get the semantics of **John runs** right, we’re going to want the extension of the constant *j* to have the same extension in all possible worlds/times.
- That is, in the translations of **John**, **Mary**, **Bill**, and **ninety** in (18d), we’re going to want *j*, *m*, *b*, *n* to behave as ‘rigid designators’.
- We can ensure this if we add the ‘meaning postulate’ below:

Meaning Postulate for Names

In a ‘logically possible’ interpretation for IL, the following formulae are true (at all worlds and times):

$$\forall x \Box[x = \alpha], \text{ where } \alpha \text{ is } j, m, b, \text{ or } n$$

*Informally:* ‘There is a (single) entity x s.t. in all possible worlds/times  $x = j$ ’  
‘There is a (single) entity x s.t. in all possible worlds/times  $x = m$ ’  
‘There is a (single) entity x s.t. in all possible worlds/times  $x = b$ ’  
‘There is a (single) entity x s.t. in all possible worlds/times  $x = n$ ’

- Thus, the truth of the formula above (at worlds/times) ensures that the constant *j* denotes the same entity in all possible worlds (and the same holds for *m*, *b*, *n*)

(23) **Remark**

Since we’ll only ever be considering ‘logically possible’ interpretations (models) for IL, we can now always regard the constants *j*, *m*, *b*, *n* as having constant intensions in IL.

- **Thus, we can now freely ‘λ-convert’ them into the scope of  $\wedge$ ,  $\Box$ , W, H**

Rule T4 handles the translation/interpretation of (positive) present-tense sentences;

Rule T17 handles the translation/interpretation of negative and past/future tense sentences...

(24) **Rule T17 (Translation Rule for Negative, Perfect, Future Sentences)**

If  $\alpha \in P_T$ ,  $\delta \in P_{IV}$ , and  $\alpha, \delta$  translate into  $\alpha'$ ,  $\delta'$  respectively, then

(i)	$F_{11}(\alpha, \delta)$	<u>translates into</u>	$\neg\alpha'([\wedge\delta'])$
(ii)	$F_{12}(\alpha, \delta)$	<u>translates into</u>	$W\alpha'([\wedge\delta'])$
(iii)	$F_{13}(\alpha, \delta)$	<u>translates into</u>	$\neg W\alpha'([\wedge\delta'])$
(iv)	$F_{14}(\alpha, \delta)$	<u>translates into</u>	$H\alpha'([\wedge\delta'])$
(v)	$F_{14}(\alpha, \delta)$	<u>translates into</u>	$\neg H\alpha'([\wedge\delta'])$

Illustrations:

- a. (i) **John translates into**  $\lambda P[P\{j\}]$ , **run translates into** **run'** (T1)
- (ii)  $F_{11}(\text{John}, \text{run})$  translates into  $\neg[\lambda P[P\{j\}]](\wedge\text{run}')$  (T17)
- (iii) **John doesn't run translates into**  $\neg[\lambda P[P\{j\}]](\wedge\text{run}')$  (def. of  $F_{11}$ )
- (iv)  $\neg[\lambda P[P\{j\}]](\wedge\text{run}') \Leftrightarrow \neg\text{run}'(j)$  ( $\lambda$ -Conv., CBN, DUC)

*Note:*

Thus, the translation of **John doesn't run** is logically equivalent to  $\neg\text{run}'(j)$

- o Thus, informally speaking, **John doesn't run** will be true at a world/time if it's false that John is running at that world/time

- b. (i) **John translates into**  $\lambda P[P\{j\}]$ , **run translates into** **run'** (T1)
- (ii)  $F_{12}(\text{John}, \text{run})$  translates into  $W[\lambda P[P\{j\}]](\wedge\text{run}')$  (T17)
- (iii) **John will run translates into**  $W[\lambda P[P\{j\}]](\wedge\text{run}')$  (def. of  $F_{12}$ )
- (iv)  $W[\lambda P[P\{j\}]](\wedge\text{run}') \Leftrightarrow W\text{run}'(j)$  ( $\lambda$ -Conv., CBN, DUC)

*Note:*

Thus, the translation of **John will run** is logically equivalent to  $W\text{run}'(j)$

- o Thus, informally speaking, **John will run** is true at a world/time if John runs at some future time at that world.

- c. (i) **John translates into**  $\lambda P[P\{j\}]$ , **run translates into** **run'** (T1)
- (ii)  $F_{15}(\text{John}, \text{run})$  translates into  $\neg H[\lambda P[P\{j\}]](\wedge\text{run}')$  (T17)
- (iii) **John hasn't run translates into**  $\neg H[\lambda P[P\{j\}]](\wedge\text{run}')$  (def. of  $F_{15}$ )
- (iv)  $\neg H[\lambda P[P\{j\}]](\wedge\text{run}') \Leftrightarrow \neg H\text{run}'(j)$  ( $\lambda$ -Conv., CBN, DUC)

*Note:*

Thus, the translation of **John hasn't run** is logically equivalent to  $\neg H\text{run}'(j)$

- o Thus, informally speaking **John hasn't run** is true at a world/time if there is no previous time at that world where John runs.

*Thus far, we've only been illustrating the compositional rules with proper names...  
Another easy set of rules to examine, though, are the translation rules for quantificational terms*

(25) **Rule T2 (Translation Rule for Quantificational Terms)**

If  $\zeta \in P_{CN}$  and  $\zeta$  translates into  $\zeta'$ , then:

- (i)  $F_0(\zeta)$  translates into  $\lambda P \Lambda x[\zeta'(x) \rightarrow P\{x\}]$
- (ii)  $F_1(\zeta)$  translates into  $\lambda P V y [\Lambda x [\zeta'(x) \leftrightarrow x = y] \& P\{y\}]$
- (iii)  $F_2(\zeta)$  translates into  $\lambda P V x[\zeta'(x) \wedge P\{x\}]$

Illustrations:

- a. (i) **man** translates into **man'** (T1)  
 (ii)  $F_0(\text{man})$  translates into  $\lambda P \Lambda x[\text{man}'(x) \rightarrow P\{x\}]$  (T2)  
 (iii) **every man** translates into  $\lambda P \Lambda x[\text{man}'(x) \rightarrow P\{x\}]$  (def. of  $F_0$ )
- b. (i) **man** translates into **man'** (T1)  
 (ii)  $F_1(\text{man})$  translates into  $\lambda P V y [\Lambda x [\text{man}'(x) \leftrightarrow x = y] \& P\{y\}]$  (T2)  
 (iii) **the man** translates into  $\lambda P V y [\Lambda x [\text{man}'(x) \leftrightarrow x = y] \& P\{y\}]$  (def. of  $F_1$ )
- c. (i) **man** translates into **man'** (T1)  
 (ii)  $F_2(\text{man})$  translates into  $\lambda P V x[\text{man}'(x) \wedge P\{x\}]$  (T2)  
 (iii) **a man** translates into  $\lambda P V x[\text{man}'(x) \wedge P\{x\}]$  (def. of  $F_2$ )

*Note:* As previewed a few classes ago, Montague adopts a ‘Russelian’ analysis of definite descriptions like “the man”.

(26) **Interactions Between Rules T2, T4, and T17**

With rules T2, T4, and T17, we can now easily translate/interpret sentences where quantificational terms occupy subject position.

- a. (i) **a man** translates into  $\lambda P V x[\text{man}'(x) \wedge P\{x\}]$  (T2)  
 (ii) **run** translates into **run'** (T1)  
 (iii)  $F_4(\text{a man}, \text{run})$  translates into  $[\lambda P V x[\text{man}'(x) \wedge P\{x\}]](^{\wedge}\text{run}')$  (T4)  
 (iv) **a man runs** translates into  $[\lambda P V x[\text{man}'(x) \wedge P\{x\}]](^{\wedge}\text{run}')$  (def. of  $F_4$ )
- (v)  $[\lambda P V x[\text{man}'(x) \wedge P\{x\}]](^{\wedge}\text{run}')$   $\Leftrightarrow$  ( $\lambda$ -Conv.)  
 (vi)  $V x[\text{man}'(x) \wedge [^{\wedge}\text{run}']\{x\}]$   $\Leftrightarrow$  (CBN)  
 (vii)  $V x[\text{man}'(x) \wedge [^{\vee}[^{\wedge}\text{run}']]])\{x\}$   $\Leftrightarrow$  (DUC)  
 (viii)  $V x[\text{man}'(x) \wedge \text{run}'(x)]$

- b. (i) **a man** translates into  $\lambda P \forall x[\mathbf{man}'(x) \wedge P\{x\}]$  (T2)
- (ii) **run** translates into **run'** (T1)
- (iii)  $F_{15}(\mathbf{a man}, \mathbf{run})$  translates into  $\neg H[\lambda P \forall x[\mathbf{man}'(x) \wedge P\{x\}]](\wedge \mathbf{run}')$  (T17)
- (iv) **a man hasn't run** translates into  $\neg H[\lambda P \forall x[\mathbf{man}'(x) \wedge P\{x\}]](\wedge \mathbf{run}')$  (def. of  $F_{15}$ )
- (v)  $\neg H[\lambda P \forall x[\mathbf{man}'(x) \wedge P\{x\}]](\wedge \mathbf{run}')$   $\Leftrightarrow$  ( $\lambda$ -Conv., CBN, DUC)
- (vi)  $\neg H \forall x[\mathbf{man}'(x) \wedge \mathbf{run}'(x)]$

(27) **Remarks**

- a. Thus, we correctly predict that **a man runs** is true at a world/time iff there is a man x (at that world/time) who runs (at that world/time)
- b. Thus, we also correctly predict that **a man hasn't run** has a ‘reading’ (translation/interpretation) that is true at a world/time iff there is no earlier time at that world where a man runs...
  - o Note that this amounts to the claim that there is a reading of **a man hasn't run** that is equivalent to **no man has run**
- c. *Of course, there is also a ‘wide-scope indefinite’ reading of **a man hasn't run**, where it is true if there is some (particular) man x such that x hasn't run.*
  - o Once we bring QI into the mix, we'll see that our fragment predicts this reading as well!

*Finally, we'll also examine the translation rules for conjunction and disjunction, since those are also relatively easy.*

(28) **Rule T11 (Conjunction and Disjunction of Sentences)**

If  $\varphi, \psi \in P_t$  and  $\varphi, \psi$  translates into  $\varphi', \psi'$  respectively, then

- (i)  $F_8(\varphi, \psi)$  translates into  $[\varphi' \wedge \psi']$
- (ii)  $F_9(\varphi, \psi)$  translates into  $[\varphi' \vee \psi']$

Illustration: The student is asked to confirm that rule T11 entails the following:

- a. The translation of **John runs and Mary talks** is logically equivalent to:  
 $[\mathbf{run}'(j) \wedge \mathbf{talk}'(m)]$
- b. The translation of **John runs or Mary talks** is logically equivalent to:  
 $[\mathbf{run}'(j) \vee \mathbf{talk}'(m)]$

(29) **Rule T12 (Conjunction and Disjunction of IVs)**

If  $\gamma, \delta \in P_{IV}$  and  $\gamma, \delta$  translate into  $\gamma', \delta'$  respectively, then

- (i)  $F_8(\gamma, \delta)$  translates into  $\lambda x[\gamma'(x) \wedge \delta'(x)]$
- (ii)  $F_9(\gamma, \delta)$  translates into  $\lambda x[\gamma'(x) \vee \delta'(x)]$

a. First Key Illustration

- (i) **run** translates into **run'**, **talk** translates into **talk'** (T1)
- (ii)  $F_8(\text{run}, \text{talk})$  translates into  $\lambda x[\text{run}'(x) \wedge \text{talk}'(x)]$  (T12)
- (iii) **run and talk** translates into  $\lambda x[\text{run}'(x) \wedge \text{talk}'(x)]$  (def. of  $F_8$ )
- (iv) **John** translates into  $\lambda P[P\{j\}]$  (T1)
- (v)  $F_4(\text{John}, \text{run and talk})$  translates into  
 $[\lambda P[P\{j\}]]([\wedge \lambda x[\text{run}'(x) \wedge \text{talk}'(x)]]])$  (T4)
- (vi) **John runs and talk** translates into  
 $[\lambda P[P\{j\}]]([\wedge \lambda x[\text{run}'(x) \wedge \text{talk}'(x)]]])$  (def. of  $F_4$ )<sup>3</sup>
- (vii)  $[\lambda P[P\{j\}]]([\wedge \lambda x[\text{run}'(x) \wedge \text{talk}'(x)]]]) \Leftrightarrow (\lambda\text{-Conv.})$
- (viii)  $[\wedge \lambda x[\text{run}'(x) \wedge \text{talk}'(x)]]\{j\} \Leftrightarrow (\text{CBN})$
- (ix)  $[\vee [\wedge \lambda x[\text{run}'(x) \wedge \text{talk}'(x)]]]\{j\} \Leftrightarrow (\text{DUC})$
- (x)  $[\lambda x[\text{run}'(x) \wedge \text{talk}'(x)]]\{j\} \Leftrightarrow (\lambda\text{-Conv.})$
- (xi)  $[\text{run}'(j) \wedge \text{talk}'(j)]$

*Note:* Thus, in our system, the translation of **John runs and talk(s)** is logically equivalent to  $[\text{run}'(j) \wedge \text{talk}'(j)]$ , which is also the translation of **John runs and John talks...**

b. Second Key Illustration:

- (i) **every man** translates into  $\lambda P \Lambda x[\text{man}'(x) \rightarrow P\{x\}]$  (T1, T2)
- (ii) **run or talk** translates into  $\lambda x[\text{run}'(x) \vee \text{talk}'(x)]$  (T1, T12)
- (iii) **every man runs or talk** translates into  
 $[\lambda P \Lambda x[\text{man}'(x) \rightarrow P\{x\}]]([\wedge \lambda x[\text{run}'(x) \vee \text{talk}'(x)]]))$  (T4)
- (iv)  $[\lambda P \Lambda x[\text{man}'(x) \rightarrow P\{x\}]]([\wedge \lambda x[\text{run}'(x) \vee \text{talk}'(x)]])) \Leftrightarrow (\lambda\text{-Conv.}, \alpha\text{-Conv.})$
- (v)  $\Lambda x[\text{man}'(x) \rightarrow [\wedge \lambda y[\text{run}'(y) \vee \text{talk}'(y)]]\{x\}] \Leftrightarrow (\text{CBN}, \text{DUC})$
- (vi)  $\Lambda x[\text{man}'(x) \rightarrow [\lambda y[\text{run}'(y) \vee \text{talk}'(y)]](x)] \Leftrightarrow (\lambda\text{-Conv.})$
- (vii)  $\Lambda x[\text{man}'(x) \rightarrow [\text{run}'(x) \vee \text{talk}'(x)]]$   
‘For all x, if x is a man, then either x runs or x talks’

*Note:* In our system, the translation of **every man runs or talk(s)** is *not* logically equivalent to the translation of **every man runs or every man talks**

---

<sup>3</sup> Note that  $F_4$  is defined so that it only inflects the first verb in a conjunction of IVs. This is clearly a problematic prediction of the PTQ system.

(30) **Rule T13 (Disjunction of Terms)**

If  $\alpha, \beta \in P_T$  and  $\alpha, \beta$  translate into  $\alpha'$ ,  $\beta'$  respectively, then  $F_9(\alpha, \beta)$  translates into  $\lambda P [\alpha'(P) \vee \beta'(P)]$

Illustration:

- (i) John translates into  $\lambda P[P\{j\}]$ , Mary translates into  $\lambda P[P\{m\}]$  (T1)
- (ii) John or Mary translates into  $\lambda P[ [\lambda P[P\{j\}]](P) \vee [ \lambda P[P\{j\}] ](P) ]$  (T13)
- (iii) John or Mary runs translates into  $[\lambda P[ [\lambda P[P\{j\}]](P) \vee [ \lambda P[P\{j\}] ](P) ]] (^{\text{run}})$  (T4)
- (iv)  $[\lambda P[ [\lambda P[P\{j\}]](P) \vee [ \lambda P[P\{j\}] ](P) ]] (^{\text{run}})$   $\Leftrightarrow$  ( $\lambda$ -Conv.)
- (v)  $[ [\lambda P[P\{j\}]](^{\text{run}}) \vee [ \lambda P[P\{j\}] ](^{\text{run}}) ]$   $\Leftrightarrow$  ( $\lambda$ -Conv.)
- (vi)  $[ [^{\text{run}}]\{j\} \vee [^{\text{run}}]\{m\} ]$   $\Leftrightarrow$  (CBN)
- (vii)  $[ [^{\vee}[^{\text{run}}]](j) \vee [^{\vee}[^{\text{run}}]](m) ]$   $\Leftrightarrow$  (DUC)
- (viii)  $[ \text{run}'(j) \vee \text{run}'(m) ]$

*Note:* Thus, our system predicts that the translation of **John or Mary runs** will be logically equivalent to the translation of **John runs or Mary runs**.

Thus far, we've covered about as much of the translation system as we can *without also discussing the translations of ‘Quantifying In’ and ‘Direct Object’ structures...*

- These will be the focus of the next handout...
- These are also, in a sense, the ‘analytic centerpiece’ of the PTQ system...

### The Proper Treatment of Quantification in Ordinary English, Part 3: The Translation System, Part 2<sup>1</sup>

*Thus far, our discussion of the translation system in PTQ has assiduously avoided transitive verbs, and the whole issue of how the contrast between seek and eat is to be approached...*

*Before we can lay out Montague's solution, however, we need to get two other items on the table.*

#### 1. Quantifying In and the *De Re / De Dicto* Ambiguity

Rule T7 below will allow us to translate/interpret sentences like **John believes that Mary runs**.

##### (1) Rule T7 (For Finite Complements of PA Verbs)

If  $\delta \in P_{IV/t}$  and  $\beta \in P_t$ , and  $\delta, \beta$  translate into  $\delta'$ ,  $\beta'$  respectively, then  $F_6(\delta, \beta)$  translates into  $\delta'(\wedge\beta')$

##### (2) Illustration of Rule T7

###### a. Obtaining the Translation

- (i) **believe that**  $\in P_{IV/t}$ , **Mary runs**  $\in P_t$  (S1, S4)
- (ii) **believe that** translates into **believe that'** (T1)
- (iii) **Mary runs** translates into  $[\lambda P[P\{m\}]](\wedge\text{run}')$  (T1, T4)
- (iv)  $F_6(\text{believe that}, \text{Mary runs})$  translates into  
 $\text{believe that}'([\wedge[\lambda P[P\{m\}]](\wedge\text{run}')])$  (T7)
- (v) **believe that Mary runs** translates into  
 $\text{believe that}'([\wedge[\lambda P[P\{m\}]](\wedge\text{run}')])$  (def. of  $F_6$ )
- (vi) **John**  $\in P_t$ , **believe that Mary runs**  $\in P_{IV}$  (S1, S7)
- (vii)  $F_4(\text{John}, \text{believe that Mary runs})$  translates into  
 $[\lambda P[P\{j\}]]([\wedge\text{believe that}'([\wedge[\lambda P[P\{m\}]](\wedge\text{run}')])])$  (T4)
- (viii) **John believes that Mary runs** translates into  
 $[\lambda P[P\{j\}]]([\wedge\text{believe that}'([\wedge[\lambda P[P\{m\}]](\wedge\text{run}')])])$  (def. of  $F_4$ )

###### b. Simplifying the Translation

- (i)  $[\lambda P[P\{j\}]]([\wedge\text{believe that}'([\wedge[\lambda P[P\{m\}]](\wedge\text{run}')])]) \Leftrightarrow (\alpha\text{-conversion})$
- (ii)  $[\lambda P[P\{j\}]]([\wedge\text{believe that}'([\wedge[\lambda Q[Q\{m\}]](\wedge\text{run}')])]) \Leftrightarrow (\lambda\text{-conversion})$
- (iii)  $[\wedge\text{believe that}'([\wedge[\lambda Q[Q\{m\}]](\wedge\text{run}')])]\{j\} \Leftrightarrow (\text{CBN, DUC})$
- (iv)  $\text{believe that}'([\wedge[\lambda Q[Q\{m\}]](\wedge\text{run}')])(j) \Leftrightarrow (\lambda\text{-conversion, CBN, DUC})$
- (v)  $\text{believe that}'([\wedge\text{run}'](m))(j)$

<sup>1</sup> These notes are based upon material in Dowty *et al.* (1981) Chapter 7.

A major component of the overall PTQ system is the translation rule for structures formed by ‘Quantifying In’ (Rules S14-S16). It’s none too different from the translation rule in our ‘toy’ PTQ system...

### (3) Rule T14 (For Translating ‘Quantifying-In’ Structures)

If  $\alpha \in P_T$ ,  $\varphi \in P_t$ , and  $\alpha, \varphi$  translate into  $\alpha'$ ,  $\varphi'$  respectively, then  $F_{10,n}(\alpha, \varphi)$  translates into  $\alpha'([\wedge \lambda x_n \varphi'])$

Note: Again, following our general pattern, in the translation of  $F_{10,n}(\alpha, \varphi)$ , the translation of  $\alpha$  takes as argument the *intension* of  $[\lambda x_n \varphi']$ .

### (3) Illustration of Rule T14

#### a. Obtaining the translation

- (i) **he<sub>3</sub>** translates into  $\lambda P[P\{x_3\}]$ , **run** translates into **run'** (T1)
- (ii) **he<sub>3</sub> runs** translates into  $\lambda P[P\{x_3\}](^{\wedge} \text{run}')$  (T4, def. of F<sub>4</sub>)
- (iii) **a man** translates into  $\lambda P Vx[\text{man}'(x) \wedge P\{x\}]$  (T2, def. of F<sub>2</sub>)
- (iv)  $F_{10,3}(\text{a man}, \text{he}_3 \text{ runs})$  translates into  
 $\lambda P Vx[\text{man}'(x) \wedge P\{x\}]([\wedge \lambda x_3[\lambda P[P\{x_3\}](^{\wedge} \text{run}')]])$  (T14)

#### b. Simplifying the Translation

- (i)  $\lambda P Vx[\text{man}'(x) \wedge P\{x\}]([\wedge \lambda x_3[\lambda P[P\{x_3\}](^{\wedge} \text{run}')]])$   $\Leftrightarrow$  ( $\alpha$ -conversion)
- (ii)  $\lambda P Vx[\text{man}'(x) \wedge P\{x\}]([\wedge \lambda x_3[\lambda Q[Q\{x_3\}](^{\wedge} \text{run}')]])$   $\Leftrightarrow$  ( $\lambda$ -conversion)
- (iii)  $Vx[\text{man}'(x) \wedge [\wedge \lambda x_3[\lambda Q[Q\{x_3\}](^{\wedge} \text{run}')]]\{x\}]$   $\Leftrightarrow$  (CBN, DUC)
- (iv)  $Vx[\text{man}'(x) \wedge [\lambda x_3[\lambda Q[Q\{x_3\}](^{\wedge} \text{run}')]](x)]$   $\Leftrightarrow$  ( $\lambda$ -conversion)
- (v)  $Vx[\text{man}'(x) \wedge [\lambda Q[Q\{x\}](^{\wedge} \text{run}')]]$   $\Leftrightarrow$  ( $\lambda$ -conversion)
- (vi)  $Vx[\text{man}'(x) \wedge [^{\wedge} \text{run}']\{x\}]$   $\Leftrightarrow$  (CBN, DUC)
- (vii)  $Vx[\text{man}'(x) \wedge \text{run}'(x)]$

With Rules T7 and T14 on the table, we can now capture the well-known ‘de re / de dicto’ ambiguity in a sentence like **John believes that a man runs.**

#### (4) The De Dicto Reading

- a. **a man runs** translates into  $[\lambda P \forall x[\text{man}'(x) \wedge P\{x\}]](\wedge \text{run}')$  (T1, T2, T4)
- b. **believe that a man runs** translates into  
 $\text{believe that}'([\wedge [\lambda P \forall x[\text{man}'(x) \wedge P\{x\}]](\wedge \text{run}'))]$  (T7)
- c. **John believes that a man runs** translates into  
 $[\lambda P[P\{j\}]]([\wedge \text{believe that}'([\wedge [\lambda P \forall x[\text{man}'(x) \wedge P\{x\}]](\wedge \text{run}'))]])$  (T4)
- d.  $[\lambda P[P\{j\}]]([\wedge \text{believe that}'([\wedge [\lambda P \forall x[\text{man}'(x) \wedge P\{x\}]](\wedge \text{run}'))])]$   
 $\Leftrightarrow (\alpha\text{-conversion}, \lambda\text{-conversion}, \text{CBN}, \text{DUC})$
- e. **believe that'**( $[\wedge [\lambda Q \forall x[\text{man}'(x) \wedge Q\{x\}]](\wedge \text{run}'))](j)$   
 $\Leftrightarrow (\lambda\text{-conversion}, \text{CBN}, \text{DUC})$
- f. **believe that'**( $[\wedge \forall x[\text{man}'(x) \wedge \text{run}'(x)]](j)$ )

Note:

Under this translation, **John believes that a man runs** ends up being true iff John stands in the **believe that'** relation to the proposition “there is a man who runs”.

- Under the auxiliary assumption that  $x$  stands in the **believe that'** relation to  $p$  iff  $p$  is true in all of  $x$ 's belief worlds, we see that this translation garners us the so-called *de dicto* reading of the sentence:
  - I.e., in all of John's belief worlds, there is a man (in that world) who runs (in that world).

#### (5) The De Re Reading, Part 1: Obtaining the Translation

- a. **he<sub>3</sub> runs** translates into  $\lambda P[P\{x_3\}](\wedge \text{run}')$  (T1, T4)
- b. **believe that he<sub>3</sub> runs** translates into  
 $\text{believe that}'([\wedge [\lambda P[P\{x_3\}](\wedge \text{run}')]])$  (T7)
- c. **John believes that he<sub>3</sub> runs** translates into  
 $[\lambda P[P\{j\}]]([\wedge \text{believe that}'([\wedge [\lambda P[P\{x_3\}](\wedge \text{run}')]])])$  (T1, T4)
- d.  $F_{10,3}(\text{a man, John believes that he}_3 \text{ runs})$  translates into  
 $[\lambda P \forall x[\text{man}'(x) \wedge P\{x\}]]$   
 $([\wedge \lambda x_3[\lambda P[P\{j\}]]([\wedge \text{believe that}'([\wedge [\lambda P[P\{x_3\}](\wedge \text{run}')]])]))$  (T2, T14)
- e. **John believes that a man runs** translates into  
 $[\lambda P \forall x[\text{man}'(x) \wedge P\{x\}]]$   
 $([\wedge \lambda x_3[\lambda P[P\{j\}]]([\wedge \text{believe that}'([\wedge [\lambda P[P\{x_3\}](\wedge \text{run}')]])]))$  (def. of  $F_{10,3}$ )

## (6) The De Re Reading, Part 2: Simplifying the Translation

Note that the ‘simplification’ below proceeds in an ‘inside-out’ fashion:

- a.  $[\lambda P \forall x [\mathbf{man}'(x) \wedge P\{x\}]]$   
 $([\wedge \lambda x_3 [\lambda P[P\{j\}]] ([\wedge \mathbf{believe that}'([\wedge [\lambda P[P\{x_3\}] (\wedge \mathbf{run}')]])])) \Leftrightarrow (\lambda\text{-conversion},$   
 $\text{CBN, DUC})$
- b.  $[\lambda P \forall x [\mathbf{man}'(x) \wedge P\{x\}]]$   
 $([\wedge \lambda x_3 [\lambda P[P\{j\}]] ([\wedge \mathbf{believe that}'([\wedge \mathbf{run}'(x_3)])]))] \Leftrightarrow (\lambda\text{-conversion},$   
 $\text{CBN, DUC})$
- c.  $[\lambda P \forall x [\mathbf{man}'(x) \wedge P\{x\}]]$   
 $([\wedge \lambda x_3 \mathbf{believe that}'([\wedge \mathbf{run}'(x_3)])(j)])] \Leftrightarrow (\lambda\text{-conversion})$
- d.  $\forall x [\mathbf{man}'(x) \wedge [\wedge \lambda x_3 \mathbf{believe that}'([\wedge \mathbf{run}'(x_3)])(j)]\{x\}] \Leftrightarrow (\text{CBN, DUC})$
- e.  $\forall x [\mathbf{man}'(x) \wedge [\lambda x_3 \mathbf{believe that}'([\wedge \mathbf{run}'(x_3)])(j)](x)] \Leftrightarrow (\lambda\text{-conversion})^2$
- f.  $\forall x [\mathbf{man}'(x) \wedge \mathbf{believe that}'([\wedge \mathbf{run}'(x)])(j)]$

Note:

Under this translation, **John believes that a man runs** ends up being true iff there is a particular man  $x$  such that John stands in the **believe that'** relation to the proposition “ $x$  runs.”

- Thus, assuming that  $y$  **believe that'**  $p$  holds iff  $p$  is true in all of  $y$ 's belief worlds, this translation garners us the so-called *de re* reading for the sentence:
  - It needn't be the case that in all (or any) of John's ‘belief worlds' there is a man running...
    - All that is required is that in all John's belief worlds,  $x$  runs, where  $x$  in the actual (evaluation) world is some man.

## (7) Remark

- Thus, the syntactic-semantic analysis in PTQ holds to the ‘scope theory’ of the *De Re* / *De Dicto* ambiguity...
- Thus, it is also subject to all the problems that we saw the scope theory is subject to in LING 620...

---

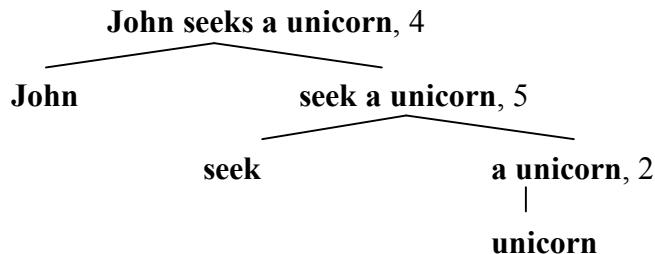
<sup>2</sup> Recall that we're able to do  $\lambda$ -conversion into the scope of ‘ $\wedge$ ’ here because ‘ $x$ ’ is a variable, and so will have the same across in all possible worlds/times.

## 2. The *De Re / De Dicto* Ambiguity with *Seeks*

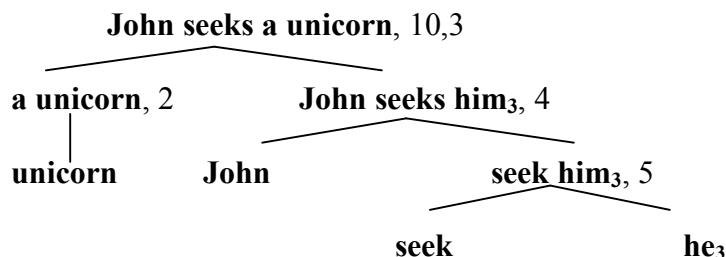
Recall that, just like the sentence **John believes that a man runs**, the sentence **John seeks a unicorn** is also syntactically ambiguous in our English fragment.

### (8) The Syntactic Ambiguity of *John seeks a unicorn*

#### a. Derivation / Analysis One



#### b. Derivation / Analysis Two



In this section, we will see that this similar syntactic ambiguity also leads to a similar semantic ambiguity:

- Under the derivation in (8a), **John seeks a unicorn** receives a *de dicto* reading (where there need be no actual unicorns)
- Under the derivation in (8b), **John seeks a unicorn** receives a *de re* reading (where there exists a particular, actual unicorn x such that John seeks x)

### (9) Rule T5 (Translation Rule for TVs and Direct Objects)

If  $\delta \in P_{TV}$  and  $\beta \in P_T$ , and  $\delta, \beta$  translate into  $\delta'$ ,  $\beta'$  respectively, then  $F_5(\delta, \beta)$  translates into  $\delta'(^{\beta'})$

Note: Again, following our general pattern, in the translation of  $F_5(\delta, \beta)$ , the translation of  $\delta$  takes as argument the *intension* of the translation of  $\beta$ .

(10) **Illustration of Rule T5 (The De Dicto Reading of *John seeks a unicorn*)**

a. *Obtaining the Translation*

- (i) **seek** translates into **seek'** (T1)
- (ii) **a unicorn** translates into  $\lambda P \forall x[\text{unicorn}'(x) \wedge P\{x\}]$  (T2)
- (iii)  $F_5(\text{seek}, \text{a unicorn})$  translates into  
 $\text{seek}'([\wedge \lambda P \forall x[\text{unicorn}'(x) \wedge P\{x\}]]])$  (T5)
- (iv) **John** translates into  $[\lambda P[P\{j\}]]$  (T1)
- (v) **John seeks a unicorn** translates into  
 $[\lambda P[P\{j\}]](\wedge \text{seek}'([\wedge \lambda P \forall x[\text{unicorn}'(x) \wedge P\{x\}]]))$  (T4)

b. *Simplifying the Translation*

- (i)  $[\lambda P[P\{j\}]](\wedge \text{seek}'([\wedge \lambda P \forall x[\text{unicorn}'(x) \wedge P\{x\}]])) \Leftrightarrow (\alpha\text{-conversion})$
- (ii)  $[\lambda Q[Q\{j\}]](\wedge \text{seek}'([\wedge \lambda P \forall x[\text{unicorn}'(x) \wedge P\{x\}]])) \Leftrightarrow (\lambda\text{-conversion})$
- (iii)  $[\wedge \text{seek}'([\wedge \lambda P \forall x[\text{unicorn}'(x) \wedge P\{x\}]])]\{j\} \Leftrightarrow (\text{CBN, DUC})$
- (iv)  $\text{seek}'([\wedge \lambda P \forall x[\text{unicorn}'(x) \wedge P\{x\}]])\{j\}$

(11) **Remarks**

- a. Note that no further simplification can take place to the formula in (10b,iv).
  - In our translation, **seek'** is a constant of type  $\langle\langle s, \langle\langle s, \langle e, t \rangle\rangle, t \rangle\rangle, \langle e, t \rangle\rangle$
  - Thus, it takes as argument (i) and (ii) to yield an expression of type  $t$ 
    - (i) the intension of a generalized quantifier expression, and
    - (ii) an entity
- b. Thus, under the syntactic derivation in (8a), **John seeks a unicorn** receives a translation which is true iff:  

$$\text{John stands in the } \text{seek}' \text{ relation to the intension of a unicorn}$$
- c. Finally, let us assume that  $x$  stands in the **seek'** relation to  $P$  iff in all the world-times  $\langle w', t' \rangle$  where  $x$ 's desires are met,  $P(w', t')(\lambda y : x \text{ has } y \text{ in } w' \text{ at } t')$ 
  - Thus, John stands in the **seek'** relation to the intension of **a unicorn** iff in all the world-times  $\langle w', t' \rangle$  where John's desires are met:  

$$\text{There is an } x \text{ such that } x \text{ is a unicorn in } w' \text{ and } t' \text{ at John has } x \text{ in } w' \text{ at } t'$$
- d. We see, then, that the translation we generate for parse (8a) amounts to the *de dicto* reading of **John seeks a unicorn**.

Now let's see what happens when we derive the sentence along the lines in (8b):

(11) **Obtaining the De Re Reading**

a. *Obtaining the Translation*

- (i) **seek** translates into **seek'**, **he<sub>3</sub>** translates into  $\lambda P[P\{x_3\}]$  (T1)
- (ii)  $F_5(\text{seek}, \text{he}_3)$  translates into **seek'**( $\wedge \lambda P[P\{x_3\}]$ ) (T5)
- (iii) **seek him<sub>3</sub>** translates into **seek'**( $\wedge \lambda P[P\{x_3\}]$ ) (def. of  $F_5$ )
- (iv) **John** translates into  $[\lambda P[P\{j\}]]$  (T1)
- (v) **John seeks him<sub>3</sub>** translates into  $[\lambda P[P\{j\}]](\wedge \text{seek}'(\wedge \lambda P[P\{x_3\}]))$  (T4)
- (vi) **a unicorn** translates into  $\lambda P Vx[\text{unicorn}'(x) \wedge P\{x\}]$  (T2)
- (vii) **John seeks a unicorn** translates into  
 $[\lambda P Vx[\text{unicorn}'(x) \wedge P\{x\}]]$   
 $(\wedge \lambda x_3[[\lambda P[P\{j\}]](\wedge \text{seek}'(\wedge \lambda P[P\{x_3\}]))])$  (T14, def. of  $F_{10,3}$ )

b. *Simplifying the Translation*

- (i)  $[\lambda P Vx[\text{unicorn}'(x) \wedge P\{x\}]](\wedge \lambda x_3[[\lambda P[P\{j\}]](\wedge \text{seek}'(\wedge \lambda P[P\{x_3\}]))])$   
 $\Leftrightarrow (\lambda\text{-conversion, CBN, DUC})$
- (ii)  $[\lambda P Vx[\text{unicorn}'(x) \wedge P\{x\}]](\wedge \lambda x_3[\text{seek}'(\wedge \lambda P[P\{x_3\}](j))])$   
 $\Leftrightarrow (\lambda\text{-conversion, CBN, DUC})$
- (iii)  $Vx[\text{unicorn}'(x) \wedge \text{seek}'(\wedge \lambda P[P\{x\}](j))]$

(12) **Remarks**

- a. Thus, under the syntactic derivation in (8b), **John seeks a unicorn** receives a translation which is true iff:  
There is an  $x$  such that  $x$  is a unicorn, and John stands in the **seek'** relation to the GQ-intension  $\wedge \lambda P[P\{x\}]$
- b. Again, let us assume that  $x$  stands in the **seek'** relation to  $\mathcal{P}$  iff in all the world-times  $\langle w', t' \rangle$  where  $x$ 's desires are met,  $\mathcal{P}(w', t')(\lambda y : x \text{ has } y \text{ in } w' \text{ at } t')$   
Thus, John stands in the **seek'** relation to  $\wedge \lambda P[P\{x\}]$  iff in all the world-times  $\langle w', t' \rangle$  such that John's desires are met, John has  $x$ .
- c. We see, then, that the translation we generate for parse (8b) amounts to the *de re* reading of **John seeks a unicorn**.

### 3. Transitive Verbs that Don't Seem to Create Opaque Contexts

#### (13) Interim Summary

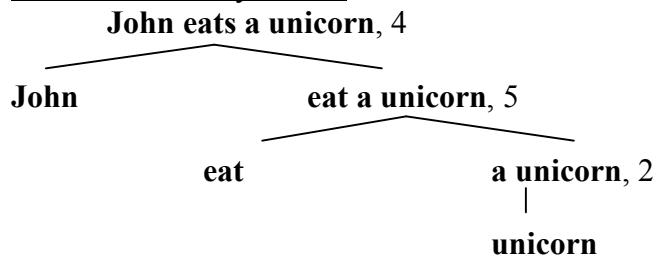
Our translation system generates two different translations for **John seeks a unicorn**

- Under one translation (reading), the sentence *does not entail that any unicorns actually exist* (the *de dicto* reading)
- Under another translation (reading), the sentence *does entail that unicorns exist*. (the *de re* reading)

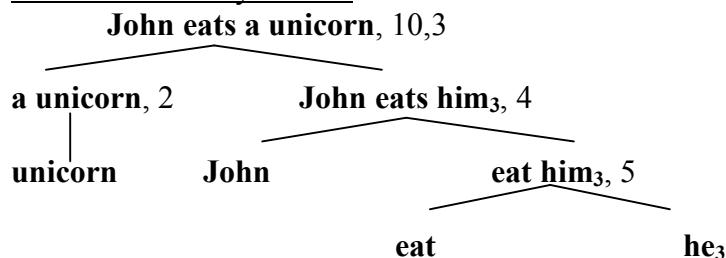
#### (14) An Immediate Problem: Parallel Ambiguity for **John eats a unicorn**.

- Note that in our English fragment, the sentence **John eats a unicorn** has the same structural ambiguity as **John seeks a unicorn**.

##### a. Derivation / Analysis One



##### b. Derivation / Analysis Two



- Furthermore, recall that – due to our category-to-type mapping – the English transitive verb **eat** is also translated as a type  $\langle\langle s, \langle\langle s, \langle e, t \rangle, t \rangle, t \rangle, e, t \rangle\rangle$  predicate.
- Consequently, our translation system predicts that **John eats a unicorn** will be ambiguous in the same way as **John seeks a unicorn**.
  - That is, **John eats a unicorn** will receive two translations, one logically equivalent to (14c), and the other equivalent to (14d).

c. Translation One:  $\text{eat}'([\wedge \lambda P Vx[\text{unicorn}'(x) \wedge P\{x\}]])(j)$

d. Translation Two:  $Vx[\text{unicorn}'(x) \wedge \text{eat}'(\wedge \lambda P[P\{x\}])](j)$

(15) **The Acute Empirical Problem: No Perception of Ambiguity**

Contrary to the predictions above, English speakers don't perceive an ambiguity in **John eats a unicorn**, akin to that in **John seeks a unicorn**.

- More acutely, under the translation in (14c), **John eats a unicorn** can be true without there being any actual unicorns (just like with (10b))
- However, English speakers universally agree that **John eats a unicorn** entails that there does exist some unicorn (which John is eating).
- That is, of the two translations in (14c,d), only (14d) seems to align with the truth-conditional judgments of English speakers.

(16) **The Key Idea Towards a Solution**

Suppose it were the case that the 'lexical semantics' of **eat** entailed that (16a) is true iff (16b) is true.

- a.  $\mathbf{eat}'(\mathcal{P})(x)$
- b.  $[\forall \mathcal{P}([\lambda x_0 [\mathbf{eat}'(\lambda P[P\{x_0\}])](x)])]$

As the computation below shows, it would predict that translation (14c) will be logically equivalent to translation (14d).

- c. (i)  $\mathbf{eat}'([\lambda P Vx[\mathbf{unicorn}'(x) \wedge P\{x\}]])(j) \Leftrightarrow (\text{by assumption (16)})$
- (ii)  $[\forall [\lambda P Vx[\mathbf{unicorn}'(x) \wedge P\{x\}]]([\lambda x_0 [\mathbf{eat}'(\lambda P[P\{x_0\}])](j)])] \Leftrightarrow (\text{CBN, DUC})$
- (iii)  $[\lambda P Vx[\mathbf{unicorn}'(x) \wedge P\{x\}]]([\lambda x_0 [\mathbf{eat}'(\lambda P[P\{x_0\}])](j)]) \Leftrightarrow (\lambda\text{-conversion})$
- (iv)  $Vx[\mathbf{unicorn}'(x) \wedge [\lambda x_0 [\mathbf{eat}'(\lambda P[P\{x_0\}])](j)]\{x\}] \Leftrightarrow (\text{CBN, DUC})$
- (v)  $Vx[\mathbf{unicorn}'(x) \wedge [\lambda x_0 [\mathbf{eat}'(\lambda P[P\{x_0\}])](j)]](x) \Leftrightarrow (\lambda\text{-conversion})$
- (vi)  $Vx[\mathbf{unicorn}'(x) \wedge \mathbf{eat}'(\lambda P[P\{x\}])](j)$

If this were the case, then, we wouldn't perceive an ambiguity in **John eats a unicorn**, since the two translations (readings) would be truth-conditionally equivalent.

- Furthermore, both translations end up entailing that *there are unicorns*.

*In the PTQ system, we have a mechanism for encoding such aspects of ‘lexical semantics’ into our analysis: the meaning postulates!*

(17) **The Solution: Meaning Postulate for ‘Non-Intensional’ Verbs**

In a ‘logically possible’ interpretation for IL, the following formulae are true (at all worlds and times):<sup>3</sup>

$$\Lambda x \Lambda \mathcal{P} \Box [ \delta(\mathcal{P})(x) \leftrightarrow [^\vee \mathcal{P}]([\wedge \lambda x_0 \delta(\wedge \lambda P[P\{x_0\}])](x))]$$

where  $\delta$  translates any member of  $B_{TV}$  other than **seek** or **conceive**

(18) **Remark**

If we restrict the interpretations of IL to only those that are ‘logically possible’, it will follow (as shown in (16)) that translation (14c) is logically equivalent to (14d).

We will also predict the univocality of such sentences as:

- |                                 |                                 |
|---------------------------------|---------------------------------|
| a. <b>John finds a unicorn.</b> | b. <b>John loses a unicorn.</b> |
| c. <b>John loves a unicorn.</b> | d. <b>John dates a unicorn.</b> |

(19) **A General Summary of the Overall Story Regarding Transitive Verbs**

- We know that **seek** must receive a translation/meaning where it takes as argument the *intension* of its complement.
- Thus, the need for a category-to-type correspondence in the PTQ (and UG) framework entails that *all* transitive verbs must take as argument the *intension* of their complement.
- Thus, even the translation/meaning of **eat** is a relation between an entity and the *intension of some GQ*.
- This, of course, raises the question of what the ‘lexical semantics’ of **eat** are. *When should we say that the eat relation holds between an entity and some GQ intension?*
- Well, suppose we say that **eat** holds between  $x$  and  $\mathcal{P}$  exactly when  $[^\vee \mathcal{P}]$  holds of the following crazy property:  $([\wedge \lambda x_0 [\text{eat}'](\wedge \lambda P[P\{x_0\}])](x))$ 
  - That’s what the meaning postulate in (17) says...
- As shown above, the resulting system correctly predicts that – while **John seeks a unicorn** doesn’t entail the existence of unicorns – **John eats a unicorn** does.

<sup>3</sup> Note that the actual formula used by Montague in PTQ is slightly different from the one in (17). However, as the student can confirm, they amount to the same condition (see Dowty *et al.* (1981), pp. 219-227).

(20) **One Final Note**

In PTQ, Montague extends the general strategy in (16)-(19) above to other categories besides TV. In this way, the PTQ system is also able to capture the observed contrasts between (i) **rapidly** vs. **allegedly**, and (ii) **in** vs. **about**.

## Final, Extended Problem Set on the PTQ System

### 1. Exercises Concerning the English Fragment

#### (1) Exercise on the Syntactic Rules of the Fragment

Please provide derivations showing that each of the following are meaningful expressions of the English fragment in PTQ.

- a. **a woman hasn't eaten the unicorn such that she has found it**
- b. **John will talk about every woman**
- c. **Bill doesn't believe that every man or every woman dates him voluntarily**

#### (2) Question Concerning Relative Clauses and Adverbs

An English sentence can contain an unbounded number of relative clauses and adverbs. Does the English fragment in PTQ accurately capture this property of English? Why or why not?

#### (3) Question Concerning Conjoined IVs

Does the English fragment in PTQ accurately predict that (3a) and (3b) are both meaningful expressions of English, while (3c) is not? Why or why not?

- a. **John wishes to walk and talk**
- b. **John walks and talks**
- c. **\* John wishes to walks and talk**

---

### 2. Exercises Concerning Intensional Logic

In the exercises below, we will assume the following meta-language abbreviations and labels for the variables and constants of IL.

Type	Variables	Constants
e	x, y, z	j, b, m
<s,e>	r	---
<e,t>	X	<b>run', man'</b>
<<s,e>,t>	Q	<b>change'</b>
<s, <e,t>>	P	---
<e,e>	----	<b>father-of'</b>
<e,<e,t>>	R	<b>love'</b>
<s,t>	p	---
<<s,t>,<e,t>>	----	<b>believe'</b>

(4) **Exercise on the Syntax of IL**

For each of the formulae below, please state whether it is or is not a meaningful expression of IL. If it is a meaningful expression, please state its type. If it is not a meaningful expression, briefly explain why.

- |                                   |   |
|-----------------------------------|---|
| a. $\wedge j$                     | f. $\lambda P P = \wedge \text{run}'$                   |
| b. $\text{change}'(\wedge j)$     | g. $[\wedge^V X](\wedge^V \wedge j)$                    |
| c. $\text{run}'(\wedge j)$        | h. $\wedge \lambda x \text{ love}'(j)(x)$               |
| d. $\lambda r \text{ change}'(r)$ | i. $\lambda p \Box p$                                   |
| e. $\lambda P \{j\}$              | j. $\lambda y \text{ father-of}'(\text{father-of}'(y))$ |

(5) **Exercise on the Semantics of IL**

Let  $\mathcal{M}$  be any intensional model  $\langle A, I, J, \leq, F \rangle$  for IL, let  $i \in I$  and  $j \in J$ , and let  $g$  be an  $\mathcal{M}$ -assignment. Prove each of the following:<sup>1</sup>

- a.  $[[ \neg W \text{ run}'(j) ]]^{M,i,j,g} = 1$  iff  
There is no  $j' \in J$  such that  $j < j'$  and  $F(\text{run}')(i,j') \wedge F(j)(i,j') = 1$
- b.  $[[ [\lambda P P \{b\}] (\text{love}'(m)) ]]^{M,i,j,g} = 1$  iff  
 $F(\text{love}')(i,j) \wedge F(m)(i,j) \wedge F(b)(i,j) = 1$
- c.  $[[ Vx [\text{man}'(x) \wedge H \text{ run}'(x)] ]]^{M,i,j,g} = 1$  iff  
there is an  $x \in D_{e,A,I,J}$  such that  $F(\text{man}')(i,j)(x) = 1$  and  
for some  $j' \in J$  such that  $j' < j$ ,  $F(\text{run}')(i,j')(x) = 1$
- d.  $[[ \text{believe}'(\wedge \text{run}'(j))(b) ]]^{M,i,j,g} = 1$  iff  
 $F(\text{believe}')(i,j)$   
(the function  $h$  with domain  $I \times J$  such that if  $i',j' \in I \times J$ , then  
 $h(i',j') = F(\text{run}')(i',j') \wedge F(j)(i',j')$ )  
 $(F(b)(i,j)) = 1$

3. **Exercises Concerning the Translation System**

(6) **Exercise on the Analysis of Conjunction and Disjunction**

Does the system in PTQ predict that (6a) and (6b) will ever receive logically equivalent translations? Is this prediction accurate? [For the purposes of this exercise, please ignore ‘quantifying in’.]

- a. **a man or a woman walks and talks.**
- b. **a man or a woman walks and a man or a woman talks**

<sup>1</sup> Note that for the purposes of this assignment, it is perfectly legitimate to show that a given formula of IL is logically equivalent to another, syntactically simpler one, and then calculate the extension of the simpler formula.

(7) **Exercise on the Analysis of De Re / De Dicto Ambiguity**

Show that the PTQ system predicts that (7a) should have a reading where it is true in scenario (7b). Is this prediction accurate?

a. The Sentence:

**John believes that a man or a woman runs.**

b. The Scenario:

John sincerely asserts “a man or the world’s best tennis player runs.” Little does John know that the world’s best tennis player is actually a woman.

---

**4. Thought Questions Concerning the MG Architecture**

(8) **Trees as Grammatical Objects**

Please consider carefully the statement below. In what ways is it accurate, and in what ways is it an (over)simplification? Please defend your answer.

“In Montague Grammar (MG), unlike in transformational frameworks like the Minimalist Program, trees are never grammatical objects. That is, in MG, the syntactic operations act on strings to create strings. Trees in MG are simply diagrams that the analyst uses as an expository devise, to compactly represent the derivations of the meaningful expressions, which are always strings.”

(9) **Truth-Conditions and Model-Theoretic Semantics**

In LING 610, we learned of the importance of *truth-conditions* to semantic theory, and that semanticists seek to build systems that pair sentences of a natural language (e.g.) English with their truth-conditions.

Do the algebraic and model-theoretic semantic systems we learned about in this class – those in UG and PTQ – do this? That is, do they make predictions regarding the *truth-conditions* of English sentences? Why or why not?