Logic-Based Natural Language Processing

Michael Kohlhase

Professur für Wissensrepräsentation und -verarbeitung
Informatik, FAU Erlangen-Nürnberg
Michael.Kohlhase@FAU.de

2024-01-20

SOME RIGHTS RESERVED

Elevator Pitch for LBS

- **Mission:** In this course we will
  - explore how to model the *meaning of natural language* via transformation into *logical systems*
  - use *logical inference* there to unravel the missing pieces; the information that is *not linguistically realized*, but is conveyed anyways.

SOME RIGHTS RESERVED

Elevator Pitch for LBS

- **Mission:** In this course we will
  - explore how to model the *meaning of natural language* via transformation into *logical systems*
  - use *logical inference* there to unravel the missing pieces; the information that is *not linguistically realized*, but is conveyed anyways.
- **Warning:** This course is only for you if you like logic, you are going to get lots of it and we are going to build our own logics, usually a new one every week or fortnight.

SOME RIGHTS RESERVED

Elevator Pitch for LBS

- **Mission:** In this course we will
  - explore how to model the *meaning of natural language* via transformation into *logical systems*
  - use *logical inference* there to unravel the missing pieces; the information that is *not linguistically realized*, but is conveyed anyways.
- **Warning:** This course is only for you if you like logic, you are going to get lots of it and we are going to build our own logics, usually a new one every week or fortnight.
- **Approach:** We will do so in a hands-on fashion using the GLIF system, formalizing NL grammars, semantics construction, and inference systems in meta-grammatical/logical systems: GF and MMT.

SOME RIGHTS RESERVED

Elevator Pitch for LBS

- **Mission:** In this course we will
  - explore how to model the *meaning of natural language* via transformation into *logical systems*
  - use *logical inference* there to unravel the missing pieces; the information that is *not linguistically realized*, but is conveyed anyways.
- **Warning:** This course is only for you if you like logic, you are going to get lots of it and we are going to build our own logics, usually a new one every week or fortnight.
- **Approach:** We will do so in a hands-on fashion using the GLIF system, formalizing NL grammars, semantics construction, and inference systems in meta-grammatical/logical systems: GF and MMT.
- **Mixing Theory and Practice:** Half of the lectures will be classroom-style teaching of theory and half will be joint formalization.

SOME RIGHTS RESERVED

Chapter 1
Administrativa

SOME RIGHTS RESERVED

Prerequisites

- **I will presuppose:** the mandatory CS courses from Semester 1-4, in particular: (or equivalent)
  - Course "Grundlagen der Logik in der Informatik" (GLOIN)
  - Course "Algorithms and data structures"
- **The following will help:** (we recap if necessary)
  - AI-1 (symbolic AI)
  - Ontologies in the semantic web (INF8)
- **Key Ingredients:** Motivation, interest, curiosity, hard work (LBS is non-trivial)
  - You can do this course if you want! (and we will help you)

SOME RIGHTS RESERVED

LBS Lab (Dogfooding our own Techniques)

- **General Plan:** We use the thursday slot to get our hands dirty with actual GLIF representations.
- **Responsible:** Frederik Schaefer (`jan.frederik.schaefer@fau.de`) Room: 11.137.
- **Goal:** Reinforce what was taught on tuesdays and have some fun.
- **Homeworks** will be small individual modeling/formalization problems (but take time to solve)
  Group submission if and only if explicitly permitted.
- **Admin:** To keep things running smoothly
  - Homeworks will be posted on course forum. (discussed in the lab)
  - Submission via StudOn (details $\leadsto$ course forum)
- **Homework Discipline:**
  - start early! (many assignments need more than one evening's work)
  - Don't start by sitting at a blank screen!
  - Humans will be trying to understand the text/code/math when grading it.

SOME RIGHTS RESERVED

Grades

- **Academic Assessment:** so far: two parts (Portfolio Assessment)
  - (20-30 min oral) or 90 min written exam at the end of the semester (50%)
  - results of the LBS lab (50%)

  This might not work with 50+ students, need to see how the course develops!
- If you have a suggestions, I will probably be happy with that as well.

SOME RIGHTS RESERVED

Textbook, Handouts and Information, Forums, Videos

- **(No) Textbook:** Course notes at `http://kwarc.info/teaching/LBS`
  - I mostly prepare them as we go along (semantically preloaded $\leadsto$ research resource)
  - Please e-mail me any errors/shortcomings you notice. (improve for group)
- **For GLIF:** Frederik's Master's Thesis [Sch20]
- **Classical Semantics/Pragmatics:** (in the FAU Library)
  - Primary reference for LBS: [CKG09] (in the FAU Library)
  - also: [HHS07; Bir13; Rie10; ZS13; Sta14; Sae03; Por04; Kea11; Jac83; Cru11; Ari10]
- **Computational Semantics:** [BB05; EU10]
- **StudOn Forum:** `https://www.studon.fau.de/crs4625835.html` for
  - announcements, homeworks (my view on the forum)
  - questions, discussion among your fellow students (your forum too, use it!)
- **Course Videos:** at `https://fau.tv/course/3647`

SOME RIGHTS RESERVED

Do I need to attend the lectures

- Attendance is not mandatory for the LBS lecture (official version)
- There are two ways of learning: (both are OK, your mileage may vary)
  - Approach B: Read a book/papers (here: course notes)
  - Approach I: come to the lectures, be involved, interrupt me whenever you have a question.

  The only advantage of I over B is that books/papers do not answer questions
- Approach S: come to the lectures and sleep does not work!
- The closer you get to research, the more we need to discuss!

SOME RIGHTS RESERVED

Experiment: Learning Support with KWARC Technologies

- **My research area:** Deep representation formats for (mathematical) knowledge
- **One Application:** Learning support systems (represent knowledge to transport it)
- **Experiment:** Start with this course (Drink my own medicine)
  1. Re-represent the slide materials in OMDoc (Open Mathematical Documents)
  2. Feed it into the ALeA system (`http://courses.voll-ki.fau.de`)
  3. Try it on you all (to get feedback from you)
- Research tasks
  - help me complete the material on the slides (what is missing/would help?)
  - I need to remember "what I say", examples on the board. (take notes)
- Benefits for you (so why should you help?)
  - you will be mentioned in the acknowledgements (for all that is worth)
  - you will help build better course materials (think of next-year's students)

SOME RIGHTS RESERVED

VoLL-KI Portal at `https://courses.voll-ki.fau.de`

- **Portal for ALeA Courses:** `https://courses.voll-ki.fau.de`

- **AI-1 in ALeA:** `https://courses.voll-ki.fau.de/course-home/ai-1`
  - All details for the course.
  - recorded syllabus (keep track of material covered in course)
  - syllabus of the last semester (for over/preview)
- **ALeA Status:** The ALeA system is deployed at FAU for over 1000 students taking six courses
  - (some) students use the system actively (our logs tell us)
  - reviews are mostly positive/enthusiastic (error reports pour in)

SOME RIGHTS RESERVED

Chapter 2
An Introduction to Natural Language Semantics

SOME RIGHTS RESERVED

Fascination of (Natural) Language

- **Definition 0.1.** A natural language is any form of spoken or signed means communication that has evolved naturally in humans through use and repetition without conscious planning or premeditation.
- **In other words:** the language you use all day long, e.g. English, German, ...
- **Why Should we care about natural language?:**
  - Even more so than thinking, language is a skill that only humans have.
  - It is a miracle that we can express complex thoughts in a sentence in a matter of seconds.
  - It is no less miraculous that a child can learn tens of thousands of words and a complex grammar in a matter of a few years.

SOME RIGHTS RESERVED

2.1 Natural Language and its Meaning

SOME RIGHTS RESERVED

What is Natural Language Semantics? A Difficult Question!

- **Question:** What is "Natural Language Semantics"?

SOME RIGHTS RESERVED

What is Natural Language Semantics? A Difficult Question!

- **Question:** What is "Natural Language Semantics"?
- **Definition 1.6 (Generic Answer).** Semantics is the study of reference, meaning, or truth.

SOME RIGHTS RESERVED

What is Natural Language Semantics? A Difficult Question!

- **Question:** What is "Natural Language Semantics"?
- **Definition 1.11 (Generic Answer).** Semantics is the study of reference, meaning, or truth.
- **Definition 1.12.** A sign is anything that communicates a meaning that is not the sign itself to the interpreter of the sign. The meaning can be intentional, as when a word is uttered with a specific meaning, or unintentional, as when a symptom is taken as a sign of a particular medical condition
  Meaning is a relationship between signs and the objects they intend, express, or signify.
- **Definition 1.13.** Reference is a relationship between objects in which one object (the name) designates, or acts as a means by which to refer to – i.e. to connect to or link to – another object (the referent).
- **Definition 1.14.** Truth is the property of being in accord with reality in a/the mind-independent world. An object ascribed truth is called true, iff it is, and false, if it is not.

SOME RIGHTS RESERVED

What is Natural Language Semantics? A Difficult Question!

- **Question:** What is "Natural Language Semantics"?
- **Definition 1.16 (Generic Answer).** Semantics is the study of reference, meaning, or truth.
- **Definition 1.17.** A sign is anything that communicates a meaning that is not the sign itself to the interpreter of the sign. The meaning can be intentional, as when a word is uttered with a specific meaning, or unintentional, as when a symptom is taken as a sign of a particular medical condition
  Meaning is a relationship between signs and the objects they intend, express, or signify.
- **Definition 1.18.** Reference is a relationship between objects in which one object (the name) designates, or acts as a means by which to refer to – i.e. to connect to or link to – another object (the referent).
- **Definition 1.19.** Truth is the property of being in accord with reality in a/the mind-independent world. An object ascribed truth is called true, iff it is, and false, if it is not.
- **Definition 1.20.** For natural language semantics, the signs are usually utterances and names are usually phrases.
- That is all very abstract and general, can we make this more concrete?
- Different (academic) disciplines find different concretizations.

SOME RIGHTS RESERVED

What is (NL) Semantics? Answers from various Disciplines!

- **Observation:** Different (academic) disciplines specialize the notion of semantics (of natural language) in different ways.

SOME RIGHTS RESERVED

What is (NL) Semantics? Answers from various Disciplines!

- **Observation:** Different (academic) disciplines specialize the notion of semantics (of natural language) in different ways.
- **Philosophy:** has a long history of trying to answer it, e.g.
  - Platon $\leadsto$ cave allegory, Aristotle $\leadsto$ Syllogisms.
  - Frege/Russell $\leadsto$ sense vs. referent. (*Michael Kohlhase* vs. *Odysseus*)

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG
SOME RIGHTS RESERVED

What is (NL) Semantics? Answers from various Disciplines!

- **Observation:** Different (academic) disciplines specialize the notion of semantics (of natural language) in different ways.
- **Philosophy:** has a long history of trying to answer it, e.g.
  - Platon $\leadsto$ cave allegory, Aristotle $\leadsto$ Syllogisms.
  - Frege/Russell $\leadsto$ sense vs. referent. (*Michael Kohlhase vs. Odysseus*)
- **Linguistics/Language Philosophy:** We need semantics e.g. in translation *Der Geist ist willig aber das Fleisch ist schwach!* vs. *Der Schnaps ist gut, aber der Braten ist verkocht!* (meaning counts)

SOME RIGHTS RESERVED

What is (NL) Semantics? Answers from various Disciplines!

- **Observation:** Different (academic) disciplines specialize the notion of semantics (of natural language) in different ways.
- **Philosophy:** has a long history of trying to answer it, e.g.
  - Platon $\leadsto$ cave allegory, Aristotle $\leadsto$ Syllogisms.
  - Frege/Russell $\leadsto$ sense vs. referent. (*Michael Kohlhase vs. Odysseus*)
- **Linguistics/Language Philosophy:** We need semantics e.g. in translation
  Der Geist ist willig aber das Fleisch ist schwach! vs.
  Der Schnaps ist gut, aber der Braten ist verkocht! (meaning counts)
- **Psychology/Cognition:** Semantics $\widehat{=}$ "what is in our brains" ($\leadsto$ mental models)

SOME RIGHTS RESERVED

What is (NL) Semantics? Answers from various Disciplines!

- **Observation:** Different (academic) disciplines specialize the notion of semantics (of natural language) in different ways.
- **Philosophy:** has a long history of trying to answer it, e.g.
  - Platon $\rightsquigarrow$ cave allegory, Aristotle $\rightsquigarrow$ Syllogisms.
  - Frege/Russell $\rightsquigarrow$ sense vs. referent. (*Michael Kohlhase vs. Odysseus*)
- **Linguistics/Language Philosophy:** We need semantics e.g. in translation *Der Geist ist willig aber das Fleisch ist schwach!* vs. *Der Schnaps ist gut, aber der Braten ist verkocht!* (meaning counts)
- **Psychology/Cognition:** Semantics $\widehat{=}$ "what is in our brains" ($\rightsquigarrow$ mental models)
- **Mathematics** has driven much of modern logic in the quest for foundations.
  - Logic as "foundation of mathematics" solved as far as possible
  - In daily practice syntax and semantics are not differentiated (much).

SOME RIGHTS RESERVED

What is (NL) Semantics? Answers from various Disciplines!

- **Observation:** Different (academic) disciplines specialize the notion of semantics (of natural language) in different ways.
- **Philosophy:** has a long history of trying to answer it, e.g.
  - Platon $\rightsquigarrow$ cave allegory, Aristotle $\rightsquigarrow$ Syllogisms.
  - Frege/Russell $\rightsquigarrow$ sense vs. referent. (*Michael Kohlhase vs. Odysseus*)
- **Linguistics/Language Philosophy:** We need semantics e.g. in translation *Der Geist ist willig aber das Fleisch ist schwach!* vs. *Der Schnaps ist gut, aber der Braten ist verkocht!* (meaning counts)
- **Psychology/Cognition:** Semantics $\hat{=}$ "what is in our brains" ($\rightsquigarrow$ mental models)
- **Mathematics** has driven much of modern logic in the quest for foundations.
  - Logic as "foundation of mathematics" solved as far as possible
  - In daily practice syntax and semantics are not differentiated (much).
- **Logic@AI/CS** tries to define meaning and compute with them. (applied semantics)
  - makes syntax explicit in a formal language (formulae, sentences)
  - defines truth/validity by mapping sentences into "world" (interpretation)
  - gives rules of truth-preserving reasoning (inference)

SOME RIGHTS RESERVED

Meaning of Natural Language; e.g. Machine Translation

- **Idea:** Machine Translation is very simple! (we have good lexica)
- **Example 1.21.** *Peter liebt Maria.* $\rightsquigarrow$ *Peter loves Mary.*
- ⚠ this only works for simple examples!
- **Example 1.22.** *Wirf der Kuh das Heu über den Zaun.* $\not\rightsquigarrow$ *Throw the cow the hay over the fence.* (differing grammar; Google Translate)
- **Example 1.23.** ⚠ Grammar is not the only problem
  - *Der Geist ist willig, aber das Fleisch ist schwach!*
  - *Der Schnaps ist gut, aber der Braten ist verkocht!*
- **Observation 1.24.** *We have to understand the meaning for high-quality translation!*

SOME RIGHTS RESERVED

Language and Information

- **Observation:** Humans use words (sentences, texts) in natural languages to represent and communicate information.
- **But:** What really counts is not the words themselves, but the meaning information they carry.

SOME RIGHTS RESERVED

Language and Information

- **Observation:** Humans use words (sentences, texts) in natural languages to represent and communicate information.
- **But:** What really counts is not the words themselves, but the meaning information they carry.
- **Example 1.27 (Word Meaning).**

Newspaper $\rightsquigarrow$ 

The New York Times
NEW YORK, THURSDAY, NOVEMBER 11, 2004
Offering Beethoven With Crisp Lines, but Softened by 'Moonlight'

- For questions/answers, it would be very useful to find out what words (sentences/texts) mean.

SOME RIGHTS RESERVED

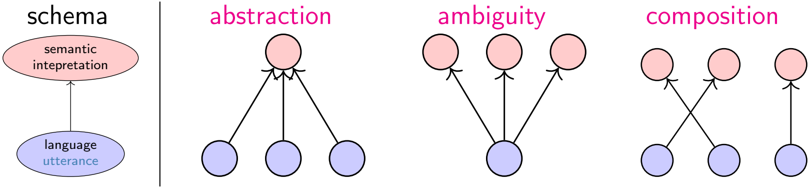
Language and Information

- **Observation:** Humans use words (sentences, texts) in natural languages to represent and communicate information.
- **But:** What really counts is not the words themselves, but the meaning information they carry.
- **Example 1.29 (Word Meaning).**

Newspaper $\rightsquigarrow$


The New York Times
NEW YORK, THURSDAY, NOVEMBER 11, 2004
RECITAL REVIEW
Offering Beethoven With Crisp Lines, but Softened by 'Moonlight'

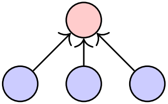
- For questions/answers, it would be very useful to find out what words (sentences/texts) mean.
- **Definition 1.30.** Interpretation of natural language utterances: three problems



SOME RIGHTS RESERVED

Language and Information (Examples)

- **Example 1.31 (Abstraction).**

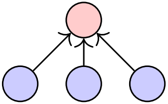


Car and *automobile* have the same meaning

SOME RIGHTS RESERVED

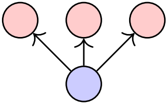
Language and Information (Examples)

- **Example 1.34 (Abstraction).**



Car and *automobile* have the same meaning

- **Example 1.35 (Ambiguity).**

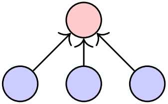


A *bank* can be a financial institution or a geographical feature

SOME RIGHTS RESERVED

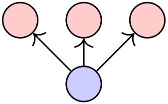
Language and Information (Examples)

- **Example 1.37 (Abstraction).**



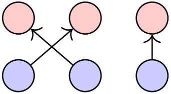
Car and *automobile* have the same meaning

- **Example 1.38 (Ambiguity).**



A *bank* can be a financial institution or a geographical feature

- **Example 1.39 (Composition).**

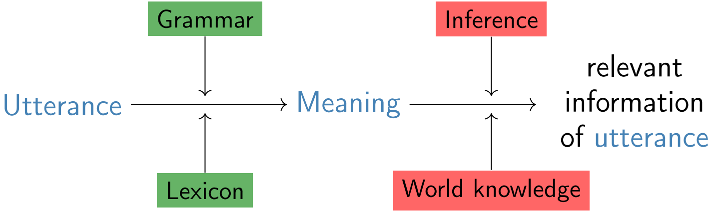


Every student sleeps $\leadsto \forall x.student(x) \Rightarrow sleep(x)$

SOME RIGHTS RESERVED

Context Contributes to the Meaning of NL Utterances

- **Observation:** Not all information conveyed is linguistically realized in an utterance.
- **Example 1.40.** *The lecture begins at 11:00 am.* What lecture? Today?
- **Definition 1.41.** We call a piece i of information linguistically realized in an utterance U, iff, we can trace i to a fragment of U.
- **Definition 1.42 (Possible Mechanism).** Inferring the missing pieces from the context and world knowledge:

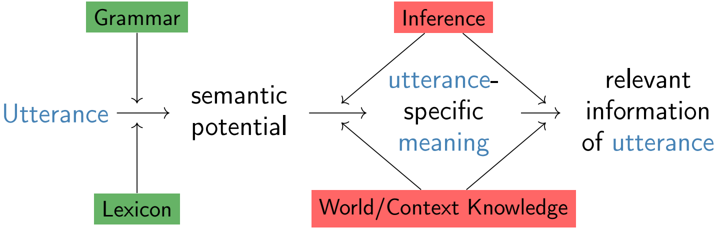


We call this process pragmatic analysis.

SOME RIGHTS RESERVED

Context Contributes to the Meaning of NL Utterances

- **Example 1.43.** *It starts at eleven.* What starts?
- Before we can resolve the time, we need to resolve the anaphor *it*.
- **Possible Mechanism:** More Inference!



$\leadsto$ Pragmatic analysis is quite complex! (prime topic of LBS)

SOME RIGHTS RESERVED

Semantics is not a Cure-It-All!

How many animals of each species did Moses take onto the ark?

SOME RIGHTS RESERVED

Semantics is not a Cure-It-All!

How many animals of each species did Moses take onto the ark?

- Actually, it was Noah (But you understood the question anyways)

SOME RIGHTS RESERVED

But Semantics works in some cases

- The only thing that currently really helps is a restricted domain:
  - I. e. a restricted vocabulary and world model.

SOME RIGHTS RESERVED

But Semantics works in some cases

- The only thing that currently really helps is a restricted domain:
  - I. e. a restricted vocabulary and world model.
- **Demo:**
  DBPedia `http://dbpedia.org/snorql/`
  Query: Soccer players, who are born in a country with more than 10 million inhabitants, who played as goalkeeper for a club that has a stadium with more than 30.000 seats and the club country is different from the birth country

SOME RIGHTS RESERVED

But Semantics works in some cases

- **Answer:**
  (is computed by DBPedia from a SPARQL query)

```
SELECT distinct ?soccerplayer ?countryOfBirth ?team ?countryOfTeam ?stadiumcapacity
{
?soccerplayer a dbo:SoccerPlayer ;
   dbo:position|dbp:position <http://dbpedia.org/resource/Goalkeeper_(association_football)> ;
   dbo:birthPlace/dbo:country* ?countryOfBirth ;
   #dbo:number 13 ;
   dbo:team ?team .
   ?team dbo:capacity ?stadiumcapacity ; dbo:ground ?countryOfTeam .
   ?countryOfBirth a dbo:Country ; dbo:populationTotal ?population .
   ?countryOfTeam a dbo:Country .
FILTER (?countryOfTeam != ?countryOfBirth)
FILTER (?stadiumcapacity > 30000)
FILTER (?population > 10000000)
} order by ?soccerplayer
```

Results: Browse Go! Reset

SPARQL results:

soccerplayer	countryOfBirth	team	countryOfTeam	stadiumcapacity
:Abdesslam_Benabdellah	:Algeria	:Wydad_Casablanca	:Morocco	67000
:Airton_Moraes_Michellon	:Brazil	:FC_Red_Bull_Salzburg	:Austria	31000
:Alain_Gouaméné	:Ivory_Coast	:Raja_Casablanca	:Morocco	67000
:Allan_McGregor	:United_Kingdom	:Beşiktaş_J.K.	:Turkey	41903
:Anthony_Scribe	:France	:FC_Dinamo_Tbilisi	:Georgia_(country)	54549
:Brahim_Zaari	:Netherlands	:Raja_Casablanca	:Morocco	67000
:Bréiner_Castillo	:Colombia	:Deportivo_Táchira	:Venezuela	38755
:Carlos_Luis_Morales	:Ecuador	:Club_Atlético_Independiente	:Argentina	48069
:Carlos_Navarro_Montoya	:Colombia	:Club_Atlético_Independiente	:Argentina	48069
:Cristián_Muñoz	:Argentina	:Colo-Colo	:Chile	47000
:Daniel_Ferreyra	:Argentina	:FBC_Melgar	:Peru	60000
:David_Bičík	:Czech_Republic	:Karşıyaka_S.K.	:Turkey	51295
:David_Loria	:Kazakhstan	:Karşıyaka_S.K.	:Turkey	51295
:Denys_Boyko	:Ukraine	:Beşiktaş_J.K.	:Turkey	41903
:Eddie_Gustafsson	:United_States	:FC_Red_Bull_Salzburg	:Austria	31000
:Emilian_Dolha	:Romania	:Lech_Poznań	:Poland	43269
:Eusebio_Acasuzo	:Peru	:Club_Bolívar	:Bolivia	42000
:Faryd_Mondragón	:Colombia	:Real_Zaragoza	:Spain	34596

SOME RIGHTS RESERVED

2.2 Natural Language Understanding as Engineering

SOME RIGHTS RESERVED

Language Technology

- Language Assistance:
  - written language: Spell/grammar/style-checking,
  - spoken language: dictation systems and screen readers,
  - multilingual text: machine-supported text and dialog translation, eLearning.

SOME RIGHTS RESERVED

Language Technology

- Language Assistance:
  - written language: Spell/grammar/style-checking,
  - spoken language: dictation systems and screen readers,
  - multilingual text: machine-supported text and dialog translation, eLearning.
- Information management:
  - search and classification of documents, (e.g. Google/Bing)
  - information extraction, question answering. (e.g. `http://ask.com`)

SOME RIGHTS RESERVED

Language Technology

- Language Assistance:
  - written language: Spell/grammar/style-checking,
  - spoken language: dictation systems and screen readers,
  - multilingual text: machine-supported text and dialog translation, eLearning.
- Information management:
  - search and classification of documents, (e.g. Google/Bing)
  - information extraction, question answering. (e.g. `http://ask.com`)
- Dialog Systems/Interfaces:
  - information systems: at airport, tele-banking, e-commerce, call centers,
  - dialog interfaces for computers, robots, cars. (e.g. Siri/Alexa)

SOME RIGHTS RESERVED

Language Technology

- Language Assistance:
  - written language: Spell/grammar/style-checking,
  - spoken language: dictation systems and screen readers,
  - multilingual text: machine-supported text and dialog translation, eLearning.
- Information management:
  - search and classification of documents, (e.g. Google/Bing)
  - information extraction, question answering. (e.g. `http://ask.com`)
- Dialog Systems/Interfaces:
  - information systems: at airport, tele-banking, e-commerce, call centers,
  - dialog interfaces for computers, robots, cars. (e.g. Siri/Alexa)
- **Observation:** The earlier technologies largely rely on pattern matching, the latter ones need to compute the meaning of the input utterances, e.g. for database lookups in information systems.

SOME RIGHTS RESERVED

What is Natural Language Processing?

- **Generally:** Studying of natural languages and development of systems that can use/generate these.
- **Definition 2.1.** Natural language processing (NLP) is an engineering field at the intersection of computer science, artificial intelligence, and linguistics which is concerned with the interactions between computers and human (natural) languages. Most challenges in NLP involve:
  - Natural language understanding (NLU) that is, enabling computers to derive meaning (representations) from human or natural language input.
  - Natural language generation (NLG) which aims at generating natural language or speech from meaning representation.
- For communication with/among humans we need both NLU and NLG.

SOME RIGHTS RESERVED

What is the State of the Art In NLU?

- Two avenues of attack for the problem: knowledge-based and statistical techniques (they are complementary)

Deep	Knowledge-based We are here	Not there yet cooperation?
Shallow	no-one wants this	Statistical Methods applications
Analysis $\uparrow$ vs. Coverage $\rightarrow$	narrow	wide

- We will cover foundational methods of deep processing in the course and a mixture of deep and shallow ones in the lab.

SOME RIGHTS RESERVED

Environmental Niches for both Approaches to NLU

- **Definition 2.2.** There are two kinds of applications/tasks in NLU:
  - Consumer tasks: consumer grade applications have tasks that must be fully generic and wide coverage. (e.g. machine translation like Google Translate)
  - Producer tasks: producer grade applications must be high-precision, but can be domain-specific (e.g. multilingual documentation, machinery-control, program verification, medical technology)

Precision

100% — Producer Tasks

50% — Consumer Tasks

$10^{3\pm1}$ Concepts $\quad$ $10^{6\pm1}$ Concepts $\quad$ **Coverage**

- **Example 2.3.** Producing/managing machine manuals in multiple languages across machine variants is a critical producer task for machine tool company.
- A producer domain I am interested in: mathematical/technical documents.

SOME RIGHTS RESERVED

NLP for NLU: The Waterfall Model

- **Definition 2.4 (The NLU Waterfall).** NL understanding is often modeled as a simple linear process: the NLU waterfall consists of five consecutive steps:
  0) speech processing: acoustic signal $\leadsto$ word hypothesis graph
  1) syntactic processing: word sequence $\leadsto$ phrase structure
  2) semantics construction: phrase structure $\leadsto$ (quasi-)logical form
  3) semantic/pragmatic analysis:
     (quasi-)logical form $\leadsto$ knowledge representation
  4) problem solving: using the generated knowledge (application-specific)
- **Definition 2.5.** We call any formalization of an utterance as a logical formula a logical form. A quasi-logical form (QLF) is a representation which can be turned into a logical form by further computation.
- **In this course:** steps 1), 2) and 3).

SOME RIGHTS RESERVED

2.3 Looking at Natural Language

SOME RIGHTS RESERVED

Fun with Diamonds (are they real?) [Dav67b]

- **Example 3.1.** We study the truth conditions of adjectival complexes:
  - *This is a diamond.* ($\models$ *diamond*)

SOME RIGHTS RESERVED

Fun with Diamonds (are they real?) [Dav67b]

- **Example 3.2.** We study the truth conditions of adjectival complexes:
  - *This is a diamond.* ($\models$ *diamond*)
  - *This is a blue diamond.* ($\models$ *diamond*, $\models$ *blue*)

SOME RIGHTS RESERVED

Fun with Diamonds (are they real?) [Dav67b]

- **Example 3.3.** We study the truth conditions of adjectival complexes:
  - *This is a diamond.* ($\models$ *diamond*)
  - *This is a blue diamond.* ($\models$ *diamond*, $\models$ *blue*)
  - *This is a big diamond.* ($\models$ *diamond*, $\not\models$ *big*)

SOME RIGHTS RESERVED

Fun with Diamonds (are they real?) [Dav67b]

- **Example 3.4.** We study the truth conditions of adjectival complexes:
  - *This is a diamond.* ($\models$ *diamond*)
  - *This is a blue diamond.* ($\models$ *diamond*, $\models$ *blue*)
  - *This is a big diamond.* ($\models$ *diamond*, $\not\models$ *big*)
  - *This is a fake diamond.* ($\models \neg$*diamond*)

SOME RIGHTS RESERVED

Fun with Diamonds (are they real?) [Dav67b]

- **Example 3.5.** We study the truth conditions of adjectival complexes:
  - *This is a diamond.* ($\models$ *diamond*)
  - *This is a blue diamond.* ($\models$ *diamond*, $\models$ *blue*)
  - *This is a big diamond.* ($\models$ *diamond*, $\not\models$ *big*)
  - *This is a fake diamond.* ($\models \neg$*diamond*)
  - *This is a fake blue diamond.* ($\models$ *blue*?, $\models$ *diamond*?)

SOME RIGHTS RESERVED

Fun with Diamonds (are they real?) [Dav67b]

- **Example 3.6.** We study the truth conditions of adjectival complexes:
  - *This is a diamond.* ($\models$ *diamond*)
  - *This is a blue diamond.* ($\models$ *diamond*, $\models$ *blue*)
  - *This is a big diamond.* ($\models$ *diamond*, $\not\models$ *big*)
  - *This is a fake diamond.* ($\models \neg$*diamond*)
  - *This is a fake blue diamond.* ($\models$ *blue*?, $\models$ *diamond*?)
  - *Mary knows that this is a diamond.* ($\models$ *diamond*)

SOME RIGHTS RESERVED

Fun with Diamonds (are they real?) [Dav67b]

- **Example 3.7.** We study the truth conditions of adjectival complexes:
  - *This is a diamond.* ($\models$ *diamond*)
  - *This is a blue diamond.* ($\models$ *diamond*, $\models$ *blue*)
  - *This is a big diamond.* ($\models$ *diamond*, $\not\models$ *big*)
  - *This is a fake diamond.* ($\models \neg$*diamond*)
  - *This is a fake blue diamond.* ($\models$ *blue?*, $\models$ *diamond?*)
  - *Mary knows that this is a diamond.* ($\models$ *diamond*)
  - *Mary believes that this is a diamond.* ($\not\models$ *diamond*)

SOME RIGHTS RESERVED

Ambiguity: The dark side of Meaning

- **Definition 3.8.** We call an utterance ambiguous, iff it has multiple meanings, which we call readings.
- **Example 3.9.** All of the following sentences are ambiguous:
  - *John went to the bank.* (river or financial?)

SOME RIGHTS RESERVED

Ambiguity: The dark side of Meaning

- **Definition 3.10.** We call an utterance ambiguous, iff it has multiple meanings, which we call readings.
- **Example 3.11.** All of the following sentences are ambiguous:
  - *John went to the bank.* (river or financial?)
  - *You should have seen the bull we got from the pope.* (three readings!)

SOME RIGHTS RESERVED

Ambiguity: The dark side of Meaning

- **Definition 3.12.** We call an utterance ambiguous, iff it has multiple meanings, which we call readings.
- **Example 3.13.** All of the following sentences are ambiguous:
  - *John went to the bank.* (river or financial?)
  - *You should have seen the bull we got from the pope.* (three readings!)
  - *I saw her duck.* (animal or action?)

SOME RIGHTS RESERVED

Ambiguity: The dark side of Meaning

- **Definition 3.14.** We call an utterance ambiguous, iff it has multiple meanings, which we call readings.
- **Example 3.15.** All of the following sentences are ambiguous:
  - *John went to the bank.* (river or financial?)
  - *You should have seen the bull we got from the pope.* (three readings!)
  - *I saw her duck.* (animal or action?)
  - *John chased the gangster in the red sports car.* (three-way too!)

SOME RIGHTS RESERVED

Quantifiers, Scope and Context

- **Example 3.16.** *Every man loves a woman.* (Keira Knightley or his mother!)

SOME RIGHTS RESERVED

Quantifiers, Scope and Context

- **Example 3.21.** *Every man loves a woman.* (Keira Knightley or his mother!)
- **Example 3.22.** *Every car has a radio.* (only one reading!)

SOME RIGHTS RESERVED

Quantifiers, Scope and Context

- **Example 3.26.** *Every man loves a woman.* (Keira Knightley or his mother!)
- **Example 3.27.** *Every car has a radio.* (only one reading!)
- **Example 3.28.** *Some student in every course sleeps in every class at least some of the time.* (how many readings?)

SOME RIGHTS RESERVED

Quantifiers, Scope and Context

- **Example 3.31.** *Every man loves a woman.* (Keira Knightley or his mother!)
- **Example 3.32.** *Every car has a radio.* (only one reading!)
- **Example 3.33.** *Some student in every course sleeps in every class at least some of the time.* (how many readings?)
- **Example 3.34.** *The president of the US is having an affair with an intern.* (2002 or 2000?)

SOME RIGHTS RESERVED

Quantifiers, Scope and Context

- **Example 3.36.** *Every man loves a woman.* (Keira Knightley or his mother!)
- **Example 3.37.** *Every car has a radio.* (only one reading!)
- **Example 3.38.** *Some student in every course sleeps in every class at least some of the time.* (how many readings?)
- **Example 3.39.** *The president of the US is having an affair with an intern.* (2002 or 2000?)
- **Example 3.40.** *Everyone is here.* (who is everyone?)

SOME RIGHTS RESERVED

More Context: Anaphora

- **Example 3.41 (Anaphoric References).**
  - *John is a bachelor. His wife is very nice.* (Uh, what?, who?)

SOME RIGHTS RESERVED

More Context: Anaphora

- **Example 3.43 (Anaphoric References).**
  - *John is a bachelor. His wife is very nice.* (Uh, what?, who?)
  - *John likes his dog Spiff even though he bites him sometimes.* (who bites?)

SOME RIGHTS RESERVED

More Context: Anaphora

- **Example 3.45 (Anaphoric References).**
  - *John is a bachelor. His wife is very nice.* (Uh, what?, who?)
  - *John likes his dog Spiff even though he bites him sometimes.* (who bites?)
  - *John likes Spiff. Peter does too.* (what to does Peter do?)

SOME RIGHTS RESERVED

More Context: Anaphora

- **Example 3.47 (Anaphoric References).**
  - *John is a bachelor. His wife is very nice.* (Uh, what?, who?)
  - *John likes his dog Spiff even though he bites him sometimes.* (who bites?)
  - *John likes Spiff. Peter does too.* (what to does Peter do?)
  - *John loves his wife. Peter does too.* (whom does Peter love?)

SOME RIGHTS RESERVED

More Context: Anaphora

- **Example 3.49 (Anaphoric References).**
  - *John is a bachelor. His wife is very nice.* (Uh, what?, who?)
  - *John likes his dog Spiff even though he bites him sometimes.* (who bites?)
  - *John likes Spiff. Peter does too.* (what to does Peter do?)
  - *John loves his wife. Peter does too.* (whom does Peter love?)
  - n*John loves golf, and Mary too.* (who does what?)
- **Definition 3.50.** A word or phrase is called anaphoric (or an anaphor), if its interpretation depends upon another phrase in context. In a narrower sense, an anaphor refers to an earlier phrase (its antecedent), while a cataphor to a later one (its postcedent).
  The process of determining the antecedent or postcedent of an anaphoric phrase is called anaphor resolution.

SOME RIGHTS RESERVED

Context is Personal and keeps changing

- *The king of America is rich.* (true or false?)

Context is Personal and keeps changing

- *The king of America is rich.* (true or false?)
- *The king of America isn't rich.* (false or true?)

SOME RIGHTS RESERVED

Context is Personal and keeps changing

- *The king of America is rich.* (true or false?)
- *The king of America isn't rich.* (false or true?)
- *If America had a king, the king of America would be rich.* (true or false!)

SOME RIGHTS RESERVED

Context is Personal and keeps changing

- *The king of America is rich.* (true or false?)
- *The king of America isn't rich.* (false or true?)
- *If America had a king, the king of America would be rich.* (true or false!)
- *The king of Buganda is rich.* (Where is Buganda?)

SOME RIGHTS RESERVED

Context is Personal and keeps changing

- *The king of America is rich.* (true or false?)
- *The king of America isn't rich.* (false or true?)
- *If America had a king, the king of America would be rich.* (true or false!)
- *The king of Buganda is rich.* (Where is Buganda?)
- *. . . Joe Smith. . . The CEO of Westinghouse announced budget cuts.*(CEO=J.S.!)

SOME RIGHTS RESERVED

2.4 A Taste of Language Philosophy

SOME RIGHTS RESERVED

What is the Meaning of Natural Language Utterances?

- **Question:** What is the meaning of the word *chair*?

SOME RIGHTS RESERVED

What is the Meaning of Natural Language Utterances?

- **Question:** What is the meaning of the word *chair*?
- **Answer:** “the set of all chairs” (difficult to delineate, but more or less clear)
- **Question:** What is the meaning of the word *Michael Kohlhase*?

SOME RIGHTS RESERVED

What is the Meaning of Natural Language Utterances?

- **Question:** What is the meaning of the word *chair*?
- **Answer:** "the set of all chairs" (difficult to delineate, but more or less clear)
- **Question:** What is the meaning of the word *Michael Kohlhase*?
- **Answer:** The word refers to an object in the real world: the instructor of LBS.
- **Alternatively:** The singleton with that object (as for "set of chairs" above)
- **Question:** What about *Michael Kohlhase sits on a chair*?

SOME RIGHTS RESERVED

What is the Meaning of Natural Language Utterances?

- **Question:** What is the meaning of the word *chair*?
- **Answer:** “the set of all chairs” (difficult to delineate, but more or less clear)
- **Question:** What is the meaning of the word *Michael Kohlhase*?
- **Answer:** The word refers to an object in the real world: the instructor of LBS.
- **Alternatively:** The singleton with that object (as for “set of chairs” above)
- **Question:** What about *Michael Kohlhase sits on a chair*?
- **Towards an Answer:** We have to combine the two sets, via the meaning of “sits”.
- **Question:** What is the meaning of the word *John F. Kennedy* or *Odysseus*?

SOME RIGHTS RESERVED

What is the Meaning of Natural Language Utterances?

- **Question:** What is the meaning of the word *chair*?
- **Answer:** "the set of all chairs" (difficult to delineate, but more or less clear)
- **Question:** What is the meaning of the word *Michael Kohlhase*?
- **Answer:** The word refers to an object in the real world: the instructor of LBS.
- **Alternatively:** The singleton with that object (as for "set of chairs" above)
- **Question:** What about *Michael Kohlhase sits on a chair*?
- **Towards an Answer:** We have to combine the two sets, via the meaning of "sits".
- **Question:** What is the meaning of the word *John F. Kennedy* or *Odysseus*?
- **Problem:** There are no objects in the real worlds, so the meaning of both is $\emptyset$ and thus equal ☹.

SOME RIGHTS RESERVED

2.4.1 Epistemology: The Philosphy of Science

SOME RIGHTS RESERVED

Epistemology – Propositions & Observations

- **Definition 4.1.** Epistemology is the branch of philosophy concerned with studying nature of knowledge, its justification, the rationality of belief, scientific theories and predictions, and various related issues.
- **Definition 4.2.** A proposition is a sentence about the actual world or a class of worlds deemed possible in a natural or formal language whose meaning can be expressed as being true or false in a specific world.
- **Definition 4.3.** A belief is a proposition φ that an agent a holds true about a class of worlds. This is a characterizing feature of the agent.
- **Definition 4.4 (Belief - The JTB Account).** Knowledge is justified, true belief.
- **Problem:** How can an agent justify a belief to obtain knowledge.
- **Definition 4.5.** Given a world w, the observed value (or just value, i.e. true or false) of a proposition (in w) can be determined by observations, that is an agent, the observer, either observes (experiences) that φ is true in w or conducts a deliberate, systematic experiment that determines φ to be true in w.

SOME RIGHTS RESERVED

Epistemology – Reproducibility & Phenomena

- **Problem:** Observations are sometimes unreliable, e.g. observer o perceives φ to be true, while it is false or vice versa.
- **Idea:** Repeat the observations to raise the probability of getting them right.
- **Definition 4.6.** An observation φ is said to be reproducible, iff φ can observed by different observers in different situations.
- **Definition 4.7.** A phenomenon φ is a proposition that is reproducibly observable to be true in a class of worlds.
- **Problem:** We would like to verify a phenomenon φ, i.e. observe φ in all worlds, But relevant world classes are too large to make this practically feasible.
- **Definition 4.8.** A world w is a counterexample to a proposition φ, if φ is observably false in w.
- **Intuition:** The absence of counterexamples is the best we can hope for in general for accepting phenomena.
- **Intuition:** The phenomena constitute the "world model" of an agent.
- **Problem:** It is impossible/inefficient (for an agent) to know all phenomena.
- **Idea:** An agent could retain only a small subset of known propositions, from this all phenomena can be derived.

SOME RIGHTS RESERVED

Epistemology – Explanations & Hypotheses

- **Definition 4.9.** A proposition ψ follows from a proposition φ, iff ψ is true in any world where φ is.
- **Definition 4.10.** An explanation of a phenomenon φ is a set Φ of propositions, such that φ follows from Φ.
- **Example 4.11.** $\{\varphi\}$ is a (rather useless) explanation for φ.
- **Intuition:** We prefer explanations Φ that explain more than just φ.
- **Observation:** This often coincides with explanations that are in some sense "simpler" or "more elementary" than φ. ($\leadsto$ Occam's razor)
- **Definition 4.12.** A proposition is called falsifiable, iff counterexamples are theoretically possible and the observation of a reproducible series of counterexample is practically feasible.
- **Definition 4.13.** A hypothesis is a proposed explanation of a phenomenon that is falsifiable.

SOME RIGHTS RESERVED

Epistemology – Scientific Theories

- **Knowledge Strategy:** Collect hypotheses about the world, drop those with counterexamples and those that can be explained themselves.
- **Definition 4.14.** A hypothesis φ can be tested in world/situation w by observing the value of φ in w. If the value is true, then we say that the observation o supports φ or is evidence for φ. If it is false then o falsifies φ.
- **Definition 4.15.** A (scientific) theory for a set Φ of phenomena is a set Θ of hypotheses that
  - has been tested extensively and rigorously without finding counterexamples, and
  - is minimal in the sense that no subset of Θ explains Φ.
- **Definition 4.16.** We call any proposition φ that follows from a theory Φ a prediction of Φ.
- **Note:** To falsify a theory Φ, it is sufficient to falsify any prediction. Any observation of a prediction φ of Φ supports Φ.

SOME RIGHTS RESERVED

2.4.2 Meaning Theories

SOME RIGHTS RESERVED

Theories of Meaning

- **The Central Question:** What is the meaning of natural language?
- This is difficult to answer definitely, ...
- **But** we can form meaning theory that make predictions that we can test.
- **Definition 4.17.** A semantic meaning theory assigns semantic contents to expressions of a language.
- **Definition 4.18.** A foundational meaning theory tries to explain why language expressions have the meanings they have; e.g. in terms of mental states of individuals and groups.
- It is important to keep these two notions apart.
- We will concentrate on semantic meaning theories in this course.

SOME RIGHTS RESERVED

The Meaning of Singular Terms

- Let's see a semantic meaning theory in action.
- **Definition 4.19.** A singular term is a phrase that purports to denote or designate a particular individual person, place, or other object.
- **Example 4.20.** *Michael Kohlhase* and *Odysseus* are singular terms.
- **Definition 4.21.** In [Fre92], Gottlob Frege distinguishes between sense (Sinn) and referent (Bedeutung) of singular terms.
- **Example 4.22.** Even though *Odysseus* does not have a referent, it has a very real sense. (but what is a sense?)
- **Example 4.23.** The ancient greeks knew the planets *Hesperos* (the evening star) and *Phosphoros* (the morning star). These words have different senses, but the – as we now know – same referent: the planet Venus.
- **Remark:** Bertrand Russell views singular terms as disguised definite descriptions – *Hesperos* as "the brightest heavenly body that sometimes rises in the evening". Frege's sense can often be conflated with Russell's descriptions. (there can be more than one definite description)

SOME RIGHTS RESERVED

Cresswell's "Most Certain Principle" and Truth Conditions

- **Problem:** How can we test meaning theories in practice?
- **Definition 4.24.** Cresswell's (1982) most certain principle (MCP): [Cre82]

  I'm going to begin by telling you what I think is the most certain thing I think about meaning. Perhaps it's the only thing. It is this. If we have two sentences A and B, and A is true and B is false, then A and B do not mean the same.
- **Definition 4.25.** The truth conditions of a sentence are the conditions of the world under which it is true. These conditions must be such that if all obtain, the sentence is true, and if one doesn't obtain, the sentence is false.
- **Observation:** Meaning determines truth conditions and vice versa.
- **In Fregean terms** The sense of a sentence (a thought) determines its referent (a truth value).

SOME RIGHTS RESERVED

a

This principle sounds trivial – and indeed it is, if you think about it – but gives rise to the notion of truth conditions, which form the most important way of finding out about the meaning of sentences: the determinations of truth conditions.

SOME RIGHTS RESERVED

Truth Conditions in Practice

- **Idea:** To test/determine the truth conditions of a sentence S in practice, we tell little stories that describe situations/worlds that embed S.
- **Example 4.26.** Consider the ambiguous sentence from 3.9
  John chased the gangster in the red sports car.
  For each of three readings there is story $\widehat{=}$ truth conditions
  - John drives the red sports car and chases the gangster
  - John chases the gangster who drives the red sports car
  - John chases the gangster on the back seat of a (very very big) red sports car.

  All of these stories correspond to different worlds, so by the MCP there must be at least three readings!

SOME RIGHTS RESERVED

Compositionality

- **Definition 4.27.** A meaning theory T is compositional, iff the meaning of an expression is a function of the meanings of its parts. We say that T obeys the compositionality principle or simply compositionality if it is.
- To compute the meaning of an expression, look up the meanings of the basic expressions forming it and successively compute the meanings of larger parts until a meaning for the whole expression is found.
- **Example 4.28 (Compositionality at work in arithmetic).** To compute the value of $(x + y)/(z \cdot u)$, look up the values of x, y, z, and u, then compute $x + y$ and $z \cdot u$, and finally compute the value of the whole expression.
- Many philosophers and linguists hold that compositionality is at work in ordinary language too.

SOME RIGHTS RESERVED

Why Compositionality is Attractive

- Compositionality gives a nice building block for a meaning theory:
- **Example 4.29.** *[Expressions [are [built [from [words [that [combine [into [[larger [and larger]] subexpressions]]]]]]]]]*
- **Consequence:** To compute the meaning of an expression, look up the meanings of its words and successively compute the meanings of larger parts until a meaning for the whole expression is found.
- Compositionality explains how people can easily understand sentences they have never heard before, even though there are an infinite number of sentences any given person at any given time has not heard before.

SOME RIGHTS RESERVED

Compositionality and the Congruence Principle

- Given reasonable assumptions compositionality entails the
- **Definition 4.30.** The congruence principle states that whenever A is part of B and A' means just the same as A, replacing A by A' in B will lead to a result that means just the same as B.
- **Example 4.31.** Consider the following (complex) sentences:
  1. *blah blah blah such and such blah blah*
  2. *blah blah blah so and so blah blah*

  If *such and such* and *so and so* mean the same thing, then 1. and 2. mean the same too.
- **Conversely:** if 1. and 2. do not mean the same, then *such and such* and *so and so* do not either.

SOME RIGHTS RESERVED

A Test for Synonymity

- Suppose we accept the most certain principle (difference in truth conditions implies difference in meaning) and the congruence principle (replacing words by synonyms results in a synonymous utterance). Then we have a diagnostics for synonymity: Replacing utterances by synonyms preserves truth conditions, or equivalently
- **Definition 4.32.** The following is called the truth conditional synonymy test:

  If replacing A by B in some sentence C does not preserve truth conditions, then A and B are not synonymous.
- We can use this as a test for the question of individuation: when are the meanings of two words the same – when are they synonymous?
- **Example 4.33 (Unsurprising Results).** The following sentences differ in truth conditions.
  1. *The cat is on the mat.*
  2. *The dog is on the mat.*

  Hence *cat* and *dog* are not synonymous. The converse holds for
  1. *John is a Greek.*
  2. *John is a Hellene.*

  In this case there is no difference in truth conditions.
- But there might be another context that does give a difference.

SOME RIGHTS RESERVED

Contentious Cases of Synonymy Test

- **Example 4.34 (Problem).** The following sentences differ in truth values:
  1. *Mary believes that John is a Greek*
  2. *Mary believes that John is a Hellene*

  So *Greek* is not synonymous to *Hellene*. The same holds in the classical example:

  1. *The Ancients knew that Hesperus was Hesperus*
  2. *The Ancients knew that Hesperus was Phosphorus*

  In these cases most language users do perceive a difference in truth conditions while some philosophers vehemently deny that the sentences under 1. could be true in situations where the 2. sentences are false.
- It is important here of course that the context of substitution is within the scope of a verb of propositional attitude. (maybe later!)

SOME RIGHTS RESERVED

A better Synonymy Test

- **Definition 4.35 (Synonymy).** The following is called the truth conditional synonymy test:

  If replacing A by B in some sentence C does not preserve truth conditions in a compositional part of C, then A and B are not synonymous.

SOME RIGHTS RESERVED

Testing Truth Conditions with Logic

- **Definition 4.36.** A logical language model $\mathcal{M}$ for a natural language L consists of a logical system $\langle\mathcal{L},\mathcal{K},\models\rangle$ and a function φ from L sentences to $\mathcal{L}$-formulae.
- **Problem:** How do we find out whether $\mathcal{M}$ models L faithfully?
- **Idea:** Test truth conditions of sentences against the predictions $\mathcal{M}$ makes.
- **Problem:** The truth conditions for a sentence S in L can only be formulated and verified by humans that speak L.
- **In Practice:** Truth conditions are expressed as "stories" that specify salient situations. Native speakers of L are asked to judge whether they make S true/false.
- **Observation 4.37.** *A logical language model $\mathcal{M}{:=}\langle L,\mathcal{L},\varphi\rangle$ can be tested:*
  1. *Select a sentence S and a situation W that makes S true.* (*according to humans*)
  2. *Translate S in to an $\mathcal{L}$-formula $S'{:=}\varphi(S)$.*
  3. *Express W as a set Φ of $\mathcal{L}$-formulae.* ($\Phi\;\widehat{=}$ *truth conditions*)
  4. *$\mathcal{M}$ is supported if $\Phi\models S'$, falsified if $\Phi\not\models S'$.*
- **Corollary 4.38.** *A logical language model constitutes a semantic meaning theory.*

SOME RIGHTS RESERVED

2.5 Computational Semantics as a Natural Science

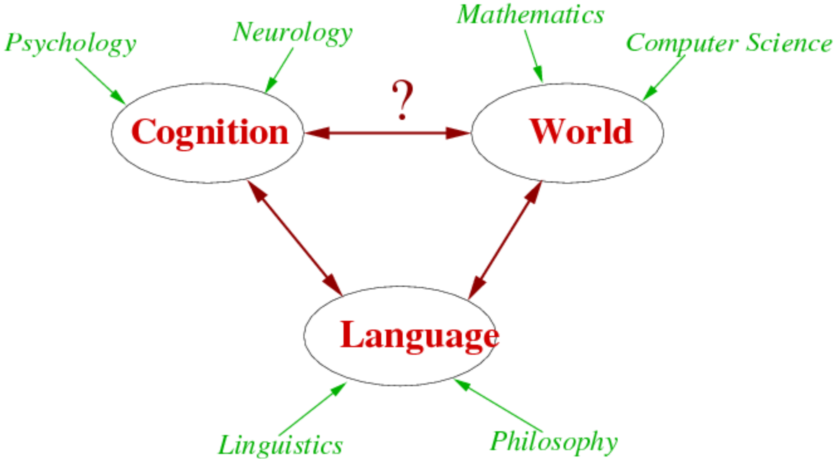
SOME RIGHTS RESERVED

Computational Semantics as a Natural Science

- **In a nutshell:** Formal logic studies formal languages, their relation with the world (in particular the truth conditions). Computational logic adds the question about the computational behavior of the relevant aspects of the formal languages.
- This is almost the same as the task of natural language semantics!
- It is one of the key ideas that logics are good scientific models for natural languages, since they simplify certain aspects so that they can be studied in isolation. In particular, we can use the general scientific method of
  1. observing
  2. building formal theories for an aspect of reality,
  3. deriving the consequences of the hypotheses about the world in the theories
  4. testing the predictions made by the theory against the real-world data. If the theory predicts the data, then this supports the theory, if not, we refine the theory, starting the process again at 2.

SOME RIGHTS RESERVED

NL Semantics as an Intersective Discipline



SOME RIGHTS RESERVED

Chapter 3
Symbolic Systems for Semantics

SOME RIGHTS RESERVED

3.1 The Grammatical Framework (GF)

SOME RIGHTS RESERVED

3.1.1 Recap: (Context-Free) Grammars

SOME RIGHTS RESERVED

Phrase Structure Grammars (Motivation)

- **Problem Recap:** We do not have enough text data to build word sequence language models $\leftsquigarrow$ data sparsity.
- **Idea:** Categorize words into classes and then generalize "acceptable word sequences" into "acceptable word class sequences" $\rightsquigarrow$ phrase structure grammars.
- **Advantage:** We can get by with much less information.
- **Example 1.1 (Generative Capacity).** 10^3 structural rules over a lexicon of 10^5 words generate most German sentences.
- Vervet monkeys, antelopes etc. use isolated symbols for sentences. $\rightsquigarrow$ restricted set of communicable propositions, no generative capacity.
- **Disadvantage:** Grammars may over generalize or under generalize.
- The formal study of grammars was introduced by Noam Chomsky in 1957 [Cho65b].

SOME RIGHTS RESERVED

Phrase Structure Grammars (cont.)

- **Example 1.2.** A simple phrase structure grammar G:

$$\begin{array}{rcl} S & \rightarrow & NP\ Vi \\ NP & \rightarrow & Article\ N \\ Article & \rightarrow & \mathrm{the} \mid \mathrm{a} \mid \mathrm{an} \\ N & \rightarrow & \mathrm{dog} \mid \mathrm{teacher} \mid \ldots \\ Vi & \rightarrow & \mathrm{sleeps} \mid \mathrm{smells} \mid \ldots \end{array}$$

  Here S, is the start symbol, NP, VP, $Article$, N, and Vi are nonterminals.
- **Definition 1.3.** The subset of lexical rules, i.e. those whose body consists of a single terminal is called its lexicon and the set of body symbols the alphabet. The nonterminals in their heads are called lexical categories.
- **Definition 1.4.** The non-lexicon production rules are called structural, and the nonterminals in the heads are called phrasal categories.

SOME RIGHTS RESERVED

Context-Free Parsing

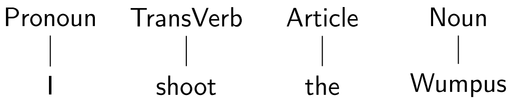
- **Recall:** The sentences accepted by a grammar are defined "top-down" as those the start symbol can be rewritten into.
- **Definition 1.5.** Bottom up parsing works by replacing any substring that matches the body of a production rule with its head.
- **Example 1.6.** Using the Wumpus grammar (below), we get the following parse trees in bottom up parsing:

I shoot the Wumpus

SOME RIGHTS RESERVED

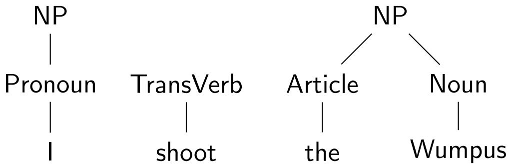
Context-Free Parsing

- **Recall:** The sentences accepted by a grammar are defined "top-down" as those the start symbol can be rewritten into.
- **Definition 1.8.** Bottom up parsing works by replacing any substring that matches the body of a production rule with its head.
- **Example 1.9.** Using the Wumpus grammar (below), we get the following parse trees in bottom up parsing:



Context-Free Parsing

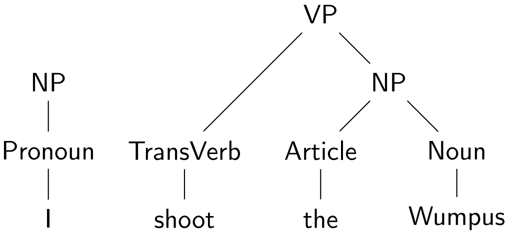
- **Recall:** The sentences accepted by a grammar are defined "top-down" as those the start symbol can be rewritten into.
- **Definition 1.11.** Bottom up parsing works by replacing any substring that matches the body of a production rule with its head.
- **Example 1.12.** Using the Wumpus grammar (below), we get the following parse trees in bottom up parsing:



SOME RIGHTS RESERVED

Context-Free Parsing

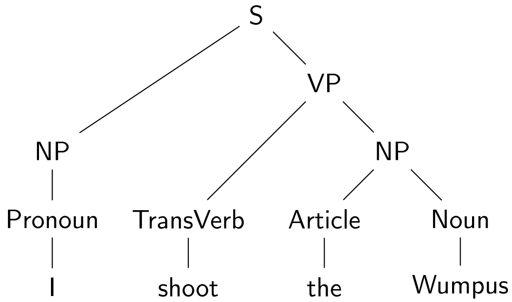
- **Recall:** The sentences accepted by a grammar are defined "top-down" as those the start symbol can be rewritten into.
- **Definition 1.14.** Bottom up parsing works by replacing any substring that matches the body of a production rule with its head.
- **Example 1.15.** Using the Wumpus grammar (below), we get the following parse trees in bottom up parsing:



SOME RIGHTS RESERVED

Context-Free Parsing

- **Recall:** The sentences accepted by a grammar are defined "top-down" as those the start symbol can be rewritten into.
- **Definition 1.17.** Bottom up parsing works by replacing any substring that matches the body of a production rule with its head.
- **Example 1.18.** Using the Wumpus grammar (below), we get the following parse trees in bottom up parsing:



Traditional linear notation: Also write this as:

$$[S[NP[Pronoun\ \text{I}]][VP[TransVerb\ \text{shoot}][NP[Article\ \text{the}][Noun\ \text{Wumpus}]]]]$$

SOME RIGHTS RESERVED

Context-Free Parsing

- **Recall:** The sentences accepted by a grammar are defined "top-down" as those the start symbol can be rewritten into.
- **Definition 1.20.** Bottom up parsing works by replacing any substring that matches the body of a production rule with its head.
- **Example 1.21.** Using the Wumpus grammar (below), we get the following parse trees in bottom up parsing:

$$[S[NP[\mathit{Pronoun}\ \mathrm{I}]][VP[\mathit{TransVerb}\ \mathrm{shoot}][NP[\mathit{Article}\ \mathrm{the}][\mathit{Noun}\ \mathrm{Wumpus}]]]]$$

- Bottom up parsing algorithms tend to be more efficient than top-down ones.
- Efficient context-free parsing algorithms run in $\mathcal{O}(n^3)$, run at several thousand words/second for real grammars.

SOME RIGHTS RESERVED

Context-Free Parsing

- **Recall:** The sentences accepted by a grammar are defined "top-down" as those the start symbol can be rewritten into.
- **Definition 1.23.** Bottom up parsing works by replacing any substring that matches the body of a production rule with its head.
- **Example 1.24.** Using the Wumpus grammar (below), we get the following parse trees in bottom up parsing:

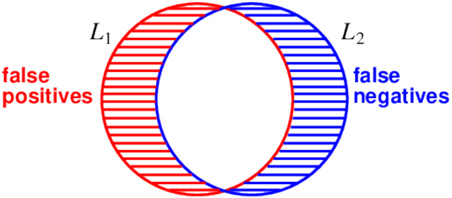
$$[S[NP[Pronoun\ \mathrm{I}]][VP[TransVerb\ \mathrm{shoot}][NP[Article\ \mathrm{the}][Noun\ \mathrm{Wumpus}]]]]$$

- Bottom up parsing algorithms tend to be more efficient than top-down ones.
- Efficient context-free parsing algorithms run in $\mathcal{O}(n^3)$, run at several thousand words/second for real grammars.
- **Theorem 1.25.** *Context-free parsing $\widehat{=}$ Boolean matrix multiplication!*
- $\leadsto$ unlikely to find faster practical algorithms. (details in [Lee02])

SOME RIGHTS RESERVED

Grammaticality Judgments

- **Problem:** The formal language $L(G)$ accepted by a grammar G may differ from the natural language L_n it supposedly models.
- **Definition 1.26.** We say that a grammar G over-generates, iff it accepts strings outside of L_n (false positives) and under-generates, iff there are L_n strings (false negatives) that $L(G)$ does not accept.



- Adjusting $L(G)$ to agree with L_n is an inductive learning problem!
  - * *the gold grab the wumpus*
  - * *I smell the wumpus the gold*
  - *I give the wumpus the gold*
  - * *I donate the wumpus the gold*
- Intersubjective agreement somewhat reliable, independent of semantics!
- Real grammars (100–5000 rules) are insufficient even for “proper” English.

SOME RIGHTS RESERVED

3.1.2 A first GF Grammar

SOME RIGHTS RESERVED

The Grammatical Framework (GF)

- **Definition 1.27.** Grammatical Framework (GF [Ran04; Ran11]) is a modular formal framework and functional programming language for writing multilingual grammars of natural languages.
- **Definition 1.28.** GF comes with the GF Resource Grammar Library, a reusable library for dealing with the morphology and syntax of a growing number of natural languages. (currently > 30)
- **Definition 1.29.** A GF grammar consists of
  - an abstract grammar that specifies well-formed abstract syntax trees (AST),
  - a collection of concrete grammars for natural languages that specify how ASTs can be linearized into (natural language) strings.
- **Definition 1.30.** Parsing is the dual to linearization, it transforms NL utterances into abstract syntax trees.
- **Definition 1.31.** The Grammatical Framwork comes with an implementation; the GF system that implements parsing, linearization, and by combination machine translation. (download/install from [GF])

SOME RIGHTS RESERVED

Hello World Example for GF (Syntactic)

- **Example 1.32 (A Hello World Grammar).**

```
abstract zero = {
  flags startcat=O;
  cat
    S ; NP ; V2 ;
  fun
    spo : V2 -> NP -> NP -> S ;
    John, Mary : NP ;
    Love : V2 ;
}
```

```
concrete zeroEng of zero = {
  lincat
    S, NP, V2 = Str ;
  lin
    spo vp s o
= s ++ vp ++ o;
    John = "John" ;
    Mary = "Mary" ;
    Love = "loves" ;
}
```

- Parse a sentence in GF: `parse "John loves Mary"` $\leadsto$ `Love John Mary`

SOME RIGHTS RESERVED

Hello World Example for GF (Syntactic)

- **Example 1.33 (A Hello World Grammar).**

```
abstract zero = {
  flags startcat=O;
  cat
    S ; NP ; V2 ;
  fun
    spo : V2 -> NP -> NP -> S ;
    John, Mary : NP ;
    Love : V2 ;
}
```

```
concrete zeroEng of zero = {
  lincat
    S, NP, V2 = Str ;
  lin
    spo vp s o
= s ++ vp ++ o;
    John = "John" ;
    Mary = "Mary" ;
    Love = "loves" ;
}
```

- Make a French grammar with `John="Jean"; Mary="Marie"; Love="aime";`
- Parse a sentence in GF: `parse "John loves Mary"` $\leadsto$ `Love John Mary`

SOME RIGHTS RESERVED

Hello World Example for GF (Syntactic)

- **Example 1.34 (A Hello World Grammar).**

```
abstract zero = {
  flags startcat=O;
  cat
    S ; NP ; V2 ;
  fun
    spo : V2 -> NP -> NP -> S ;
    John, Mary : NP ;
    Love : V2 ;
}
```

```
concrete zeroEng of zero = {
  lincat
    S, NP, V2 = Str ;
  lin
    spo vp s o
= s ++ vp ++ o;
    John = "John" ;
    Mary = "Mary" ;
    Love = "loves" ;
}
```

- Make a French grammar with `John="Jean"; Mary="Marie"; Love="aime";`
- Parse a sentence in GF: `parse "John loves Mary"` $\leadsto$ `Love John Mary`
- Linearize in GF: `linearize Love John Mary` $\leadsto$ `John loves Mary`
- translate in GF:
  `parse -lang=Eng "John Loves Mary" | linearize -lang=Fre`
- generate random sentences to test:
  `generate_random -number=10 | linearize -lang=Fre` $\leadsto$
  `Jean aime Marie`

SOME RIGHTS RESERVED

Translation to Logic

- **Idea:** Use logic as a "natural language" (to translate into)
- **Example 1.35 (Hello Prolog).** Linearize to Prolog terms:

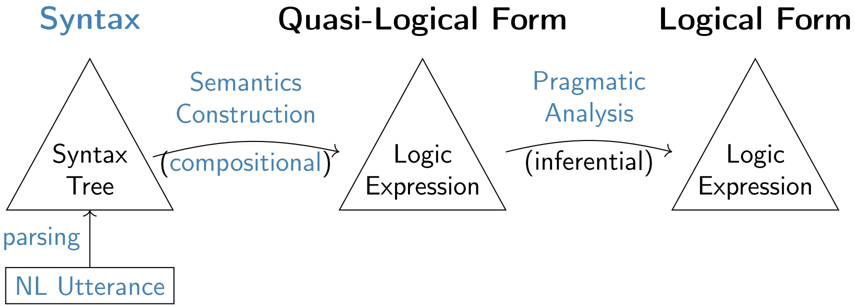
```
concrete zeroPro of zero = {
  lincat
    S , NP , V2 = Str;
  lin
    spo = \vt,subj,obj -> vt ++ "(" ++ subj ++ "," ++ obj ++ ").";
    John = "john";
    Mary = "mary";
    Love = "loves";
  }
```

- **Linearization in GF:** `linearize Love John Mary` $\rightsquigarrow$ `loves ( john , mary )`
- **Note:** `loves ( john , mary )` is *not* a quasi-logical forms, but a Prolog term that can be read into an Prolog interpreter for pragmatic analysis.

SOME RIGHTS RESERVED

Syntactic and Semantic Grammars

- Recall our interpretation pipeline

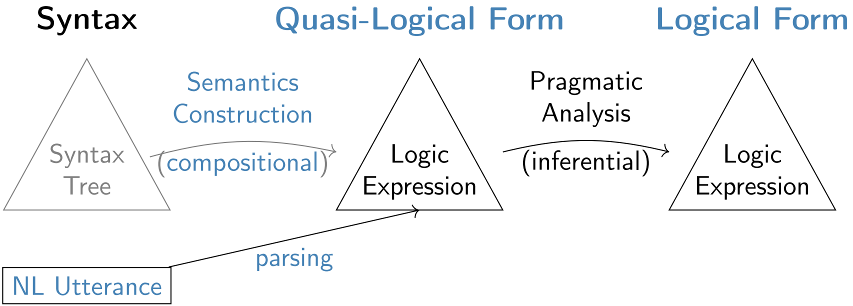


- **Definition 1.36.** We call a grammar syntactic, iff the categories and constructors are motivated by the syntactic structure of the utterance, and semantic, iff they are motivated by the structure of the domain to be modeled.
- Grammar `zero` from 1.32 is syntactic.

SOME RIGHTS RESERVED

Syntactic and Semantic Grammars

- Recall our interpretation pipeline



- **Definition 1.37.** We call a grammar syntactic, iff the categories and constructors are motivated by the syntactic structure of the utterance, and semantic, iff they are motivated by the structure of the domain to be modeled.
- Grammar `zero` from 1.32 is syntactic.
- We will look at semantic versions next.

Hello World Example for GF (semantic)

- A semantic Hello World Grammar

```
abstract one = {
  flags startcat = O;
  cat
    I; -- Individuals
    O; -- Statements
  fun
    John, Mary : I;
    Love : I -> I -> O;
}
```

```
concrete oneEng of one = {
  lincat
    I = Str ;
    O = Str ;
  lin
    John = "John";
    Mary = "Mary";
    Love s o = s ++ "loves" ++ o;
  }
```

- Instead of the "syntactic categories" `S` (sentence), `NP` (noun phrase), and `V2` (transitive verb), we now have the semantic categories `I` (individual) and `O` (proposition).

SOME RIGHTS RESERVED

3.1.3 Inflection and Case in GF

SOME RIGHTS RESERVED

Towards Complex Linearizations (Setup/English)

- Extending our hello world grammar(the trivial bit) We add the determiner *the* as an operator that turns a noun (N) into a noun phrase (NP)

```
abstract two = {
  flags startcat=O;
  cat
    S ; NP ; V2 ; N;
  fun
    spo : V2 -> NP -> NP -> S ;
    John, Mary : NP ;
    Love : V2 ;
    dog, mouse : N;
    the : N -> NP ;
}
```

```
concrete twoEN of two = {
  lincat
    S, NP, V2, N = Str ;
  lin
    spo vp s o
= s ++ vp ++ o;
    John = "John" ;
    Mary = "Mary" ;
    Love = "loves" ;
    dog = "dog" ;
    mouse = "mouse" ;
    the x = "the" ++ x;
}
```

- **Idea:** A noun phrase is a phrase that can be used wherever a proper name can be used.

SOME RIGHTS RESERVED

Towards Complex Linearizations (German)

- We try the same for German

```
abstract two = {
  flags startcat=O;
  cat
    S ; NP ; V2 ; N;
  fun
    spo : V2 -> NP -> NP -> S ;
    John, Mary : NP ;
    Love : V2 ;
    dog, mouse : N;
    the : N -> NP ;
}
```

```
concrete twoDE0 of two = {
  lincat S, NP, V2, N = Str ;
  lin
    spo vp s o = s ++ vp ++ o;
    John = "Johann" ;
    Mary = "Maria" ;
    Love = "liebt" ;
    dog = "Hund" ;
    mouse = "Maus" ;
    the x = "der" ++ x;
}
```

- Let us test-drive this; as expected we obtain

```
two> l -lang=DE0 spo Love John (the dog)
Johann liebt der Hund
```

- **Problem:** *Johann liebt der Hund* is not grammatical in German
  $\leadsto$ We need to take (grammatical) gender into account to obtain the correct form *den* of the determiner.

SOME RIGHTS RESERVED

Adding Gender

- To add gender, we add a parameter and extend the type `N` to a record

```
concrete twoDE1 of two = {
  param
    Gender = masc | fem | neut;
  lincat
    S, V2, NP = Str ;
    N = {s : Str; gender : Gender};
  lin
    spo vp s o = s ++ vp ++ o;
    John = "Johann" ;
    Mary = "Maria" ;
    Love = "liebt" ;
    dog = {s = "Hund"; gender = masc} ;
    mouse = {s = "Maus" ; gender = fem} ;
    the x = case x.gender of {masc => "der" ++ x.s;
                              fem => "die" ++ x.s;
                              neut => "das" ++ x.s} ;
}
```


SOME RIGHTS RESERVED

Adding Gender

- Let us test-drive this; as expected we obtain

```
two> l -lang=DE1 spo Love (the mouse) Mary
Die Maus liebt Maria.
```

```
two> l -lang=DE1 spo Love Mary (the dog)
Maria liebt der Hund.
```

- We need to take into account case in German too.

SOME RIGHTS RESERVED

Adding Case

- To add case, we add a parameter, reinterpret type NP as a case-dependent table of forms.

```
concrete twoDE2 of two = {
  param
    Gender = masc | fem | neut;
    Case = nom | acc;
  lincat
    S, V2 = {s: Str} ;
    N = {s : Str; gender : Gender};
    NP = {s : Case => Str};
```


SOME RIGHTS RESERVED

Adding Case

```
lin
  spo vp subj obj = {s = subj.s!nom ++ vp.s ++ obj.s!acc};
  John = {s = table {nom => "Johann"; acc => "Johann"}};
  Mary = {s = table {nom => "Maria"; acc => "Maria"}};
  Love = {s = "liebt"} ;
  dog = {s = "Hund"; gender = masc} ;
  mouse = {s = "Maus" ; gender = fem} ;
  the x = {s = table
             { nom => case x.gender of {masc => "der" ++ x.s;
                                        fem => "die" ++ x.s;
                                        neut => "das" ++ x.s};
               acc => case x.gender of {masc => "den" ++ x.s;
                                        fem => "die" ++ x.s;
                                        neut => "das" ++ x.s}}};}
```

- Let us test-drive this; as expected we obtain

```
two> l -lang=DE2 spo Love Mary (the dog)
Maria liebt den Hund.
```


SOME RIGHTS RESERVED

Adding Operations (reusable components)

- We add operations (functions with $\lambda \mathrel{\widehat{=}}$ \) to get the final form.

```
concrete twoDE of two = {
  param
    Gender = masc | fem | neut;
    Case = nom | acc;
  oper
    Noun : Type = {s : Str; gender : Gender};

    mkPN : Str −> NP = \x −> lin NP {s = table {nom => x; acc => x}};
    mkV2 : Str −> V2 = \x −> lin V2 {s = x};
    mkN : Str −> Gender −> Noun = \x,g −> {s = x; gender = g};
    mkXXX : Str −> Str −> Str −> Noun −> Str =
      \ma,fe,ne,noun −> case noun.gender of {masc => ma ++ noun.s;
                                              fem => fe ++ noun.s;
                                              neut => ne ++ noun.s};
```


SOME RIGHTS RESERVED

Adding Operations (reusable components)

```
lincat
    S, V2 = {s : Str};
    N = Noun;
    NP = {s: Case => Str};
lin
    spo vp subj obj = {s = subj.s!nom ++ vp.s ++ obj.s!acc};
    John = mkPN "Johannes";
    Mary = mkPN "Maria";
    Love = mkV2 "liebt";
    dog = mkN "Hund" masc;
    mouse = mkN "Maus" fem;
    the n = {s = table { nom => mkXXX "der" "die" "das" n;
                         acc => mkXXX "den" "die" "das" n}
    };
}
```


SOME RIGHTS RESERVED

3.1.4 Engineering Resource Grammars in GF

SOME RIGHTS RESERVED

Modular Grammars (Abstract)

- We split the grammar into modules (resource + application grammar)

Monolithic	Modular

abstract two = {
flags startcat=O;
cat
S ; NP ; V2 ; N;
fun
spo : V2 −> NP −> NP −> S ;
John, Mary : NP ;
Love : V2 ;
dog, mouse : N;
the : N −> NP ;
}
	abstract twoCat = {
	cat S ; NP ; V2 ; N;}
	abstract twoGrammar = twoCat ∗∗ {
	fun
	spo : V2 −> NP −> NP −> S ;
	the : N −> NP ; }
	abstract twoLex = twoCat ∗∗ {
	fun
	John, Mary : NP ;
	Love : V2 ;
	dog, mouse : N;}
	abstract twoRG = twoGrammar,twoLex;
	∗∗ {**flags** startcat=O;}

- Functionality is the same, but we can reuse the components

Modular Grammars (Concrete English)

- We split the grammar into modules (resource + application grammar)

Monolithic	Modular

concrete twoEN **of** two = {
  lincat
    S, NP, V2, N = Str ;
  lin
    spo vp s o = s ++ vp ++ o;
    John = *"John"* ;
    Mary = *"Mary"* ;
    Love = *"loves"* ;
    dog = *"dog"* ;
    mouse = *"mouse"* ;
    the x = *"the"* ++ x;
}
 	
resource twoParadigmsEN =
  twoCatEN ** {**oper**
    mkPN : Str -> StringType
      = \x -> {s = x};
    mkV2 : Str -> StringType
      = \x -> {s = x};
    mkN : Str -> StringType
      = \x -> {s = x};}
	concrete twoCatEN **of** twoCat = {
	  oper StringType : Type = {s : Str};
	  lincat
	    S, NP, N, V2 = StringType ;}
	
	concrete twoGrammarEN **of** twoGrammar =
	  twoCatEN ** {
	  lin
	    spo vp s o
	= {s= s.s ++ vp.s ++ o.s};
	    the x = {s = *"the"* ++ x.s};}
	
	concrete twoLexEN **of** twoLex =
	  twoCatEN ** **open** twoParadigmsEN **in** {
	  lin
	    John = mkPN *"John"* ;
	    Mary = mkPN *"Mary"* ;
	    Love = mkV2 *"loves"* ;
	    dog = mkN *"dog"* ;
	    mouse = mkN *"mouse"* ;}
	
	concrete twoRGEN **of** twoRG =
	  twoGrammarEN,twoLexEN;

SOME RIGHTS RESERVED

Modular Grammars (Concrete German)

- We split the grammar into modules (resource + application grammar)

```
concrete twoCatDE of twoCat = {
  param
    Gender = masc | fem | neut;
    Case = nom | acc;
  oper
    Noun : Type = {s : Str; gender : Gender};
    NounPhrase : Type = {s: Case => Str};
  lincat
    S, V2 = {s : Str};
    N = Noun;
    NP = NounPhrase;}

resource twoParadigmsDE = twoCatDE ** {
  oper
    mkPN : Str -> NounPhrase = \x -> {s = table {nom => x; acc => x}};
    mkV2 : Str -> V2 = \x -> lin V2 {s = x};
    mkN : Str -> Gender -> Noun = \x,g -> {s = x; gender = g};
    mkXXX : Str -> Str -> Str -> Noun -> Str =
      \ma,fe,ne,noun -> case noun.gender of {masc => ma ++ noun.s;
                                             fem => fe ++ noun.s;
                                             neut => ne ++ noun.s};}
```


SOME RIGHTS RESERVED

Modular Grammars (Concrete German)

- ```
  concrete twoGrammarDE of twoGrammar =
    twoCatDE ** open twoParadigmsDE in {
    lin
      spo vp subj obj = {s = subj.s!nom ++ vp.s ++ obj.s!acc};
      the n = {s = table { nom => mkXXX "der" "die" "das" n;
                           acc => mkXXX "den" "die" "das" n}};}

  concrete twoLexDE of twoLex = twoCatDE ** open twoParadigmsDE in {
    lin
      John = mkPN "Johannes";
      Mary = mkPN "Maria";
      Love = mkV2 "liebt";
      dog = mkN "Hund" masc;
      mouse = mkN "Maus" fem;}

  concrete twoRGDE of twoRG = twoGrammarDE,twoLexDE;
  ```


SOME RIGHTS RESERVED
  ```

A Semantic Grammar

- We use logic-inspired categories instead of the syntactic ones

Syntactic	Semantic

abstract two = {
  flags startcat=O;
  cat
    S ; NP ; V2 ; N;
  fun
    spo : V2 −> NP −> NP −> S ;
    John, Mary : NP ;
    Love : V2 ;
    dog, mouse : N;
    the : N −> NP ;
}
	abstract three = {
	  flags startcat=O;
	  cat
	    I; O; P1; P2;
	  fun
	    spo : P2 −> I −> I −> O ;
	    John, Mary : I ;
	    Love : P2 ;
	    dog, mouse : P1;
	    the : P1 −> I;
	}

SOME RIGHTS RESERVED

A Semantic Grammar (Modular Development)

- We use logic-inspired categories instead of the syntactic ones

Syntactic	Semantic

concrete twoCatEN **of** twoCat = {
  oper StringType : Type = {s : Str};
  lincat
    S, NP, N, V2 = StringType ;}
 	
concrete twoGrammarEN **of** twoGrammar =
  twoCatEN ∗∗ {
  lin
    spo vp s o = {s= s.s ++ vp.s ++ o.s};
    the x = {s = *"the"* ++ x.s};}
 	
concrete twoLexEN **of** twoLex =
  twoCatEN ∗∗ **open** twoParadigmsEN **in** {
  lin
    John = mkPN *"John"* ;
    Mary = mkPN *"Mary"* ;
    Love = mkV2 *"loves"* ;
    dog = mkN *"dog"* ;
    mouse = mkN *"mouse"* ;}
 	
concrete twoRGEN **of** twoRG =
  twoGrammarEN,twoLexEN;
	concrete threeEN **of** three =
	  twoLexEN,twoGrammarEN ∗∗
	  open twoParadigmsEN **in** {
	  lincat
	    I = NP;
	    O = S;
	    P1 = N;
	    P2 = V2;
	}
	
	concrete threeDE **of** three =
	  twoLexDE,twoGrammarDE ∗∗
	  open twoParadigmsDE **in** {
	  lincat
	    I = NP;
	    O = S;
	    P1 = N;
	    P2 = V2;
	}

SOME RIGHTS RESERVED

3.2 MMT: A Modular Framework for Representing Logics and Domains

SOME RIGHTS RESERVED

3.2.1 Propositional Logic in MMT: A first Example

SOME RIGHTS RESERVED

Implementing minimal PL^0 in Mmt

- **Recall:** The language $\mathit{wff}_0(\Sigma_0)$ of propositional logic (PL^0) consists of propositions built from propositional variables from $\mathcal{V}_0$ and connectives from Σ_0.
- We model $\mathit{wff}_0(\Sigma_0)$ in a Mmt theory ($\Sigma_0 := \{\neg, \wedge\}$ for the moment)

```
theory proplogMinimal : ur:?LF =
▮
```

  - theory is the Mmt keyword for modules, the module delimiter ▮ delimits them.
  - A theory has a local name and a meta-theory (after the :)
    Here it is LF (provides the logical constants $\rightarrow$, *type*, λ, Π)
- Mmt theories contain declarations of the form
  ⟪name⟫ : ⟪type⟫ |# ⟪notation⟫
  - declarations are delimited by the declaration delimiter ▮,
  - declaration components by the object delimiter |.
- **Example 2.1.** A declaration for the type of propositions

```
    prop : type |# o ▮
```

  - the local name prop is the system identifier
  - the type type declares prop to be a type (optional part)
  - the notation definition o declares the notation for prop (can be used instead) (optional part)

SOME RIGHTS RESERVED

Implementing minimal PL^0 in Mmt (continued)

- **Example 2.2.** Declarations for the connectives ¬ and ∧

```
not : o → o | # ¬1 prec 100 ❚
```

  - the type o → o declares the constant not to be a unary function
  - the notation definition ¬1 prec 100 establishes
    - the function symbol ¬ for not followed by argument 1.
    - brackets are governed by the precedence 100 (binding strength)

```
and : o → o → o | # 1 ∧ 2 prec 90❚
```

  - The type o → o → o declares the constant and to be a binary function (note currying)
  - the notation definition # 1 ∧ 2 prec 90 establishes
    - the infix function symbol ∧ for and preceded by argument 1 and followed by 2,
    - brackets are governed by the precedence 90 (weaker than for not)
  - **Testing precedences:** the Mmt system accepts A : o ❚ test : ¬A ∧ A ❚
    And ¬A ∧ A is parsed as (¬A) ∧ A instead of ¬(A ∧ A)
- **All together now! PL^0 Syntax as a Mmt theory:**

```
theory proplogMinimal : ur:?LF =
    prop : type | # o ❚
    not : o → o | # ¬1 prec 100 ❚
    and : o → o → o | # 1 ∧ 2 prec 90❚
❚
```


SOME RIGHTS RESERVED

Completing PL^0 by Definitions

- Building on this, we can define additional connectives: $\vee$, $\Rightarrow$, $\Leftrightarrow$

```
theory proplog : ur:?LF =
    include ?proplogMinimal ❚
    or : o → o → o | # 1 ∨ 2 prec 80 | = [a:o,b:o] ¬(¬ a ∧ ¬b) ❚
    implies : o → o → o | # 1 ⇒ 2 prec 70 | = [a:o,b:o] ¬a ∨ b ❚
❚
```

  - `include` is the keyword for an inclusion declaration
    here we include the theory `proplogMinimal` (notation: theory refs prefixed by `?`)
    this makes all of its declarations available locally in theory `proplog`.
  - new declaration components: definientia give a constant meaning by replacement.
  - `[a:o,b:o] ¬a ∨ b` is the Mmt notation for $\lambda a_o b_o . \neg a \vee b$, i.e. the function that given two propositions a and b returns the proposition $\neg a \vee b$.
  - **Note**: types optional in lambdas (Mmt system infers them from context)
- This completes the syntax (language of formulae) of PL^0.
- **Observation:** The declarations in `proplog` amount to a context-free grammar of PL^0.

SOME RIGHTS RESERVED

Describing Situations for Truth Conditions

- We want to derive the truth conditions e.g. for *Peter loves Mary*.
- **Definition 2.3.** A situation theory is an Mmt theory that formalizes a situation.
- **First Attempt:** We provide declarations for the individuals and their relations.

```
theory world1 : ur:?LF =
    include ?proplog ❚

    individual : type ❘# ι ❚
    peter : ι ❚
    mary : ι ❚
    loves : ι → ι → o ❚

    plm = loves peter mary ❚// just an abbreviation ❚
❚
```

- **Problem:** We have not asserted that plm is true in world1, ...
  ...only that the proposition plm exists.
- **Idea:** Let's assert that plm is "provable" in theory world1.

SOME RIGHTS RESERVED

Asserting Truth by Declaring Provability in Mmt Theories

- **Observation:** We can only assert existance in a theory by declarations.
- **Idea 1:** Use declarations to declare certain types to be inhabited $\widehat{=}$ non-empty.
- **Idea 2:** A proposition A is "provable", iff the "type of all proofs of A" is inhabited.
- **Idea 3:** We can express "the type of all proofs of A" as $\vdash A$ if we declare a suitable type constructor in Mmt:

  ```
  ded : prop → type | # ⊢1
  ```

- **All Together Now:** We can assert that *Peter loves Mary* in theory `world1`

  ```
  plm_axiom : ⊢plm // the type of proofs of plm is inhabited
  ```

  Note that in this interpretation the constant `plm_axiom` is a "proof of `plm`"
- **Definition 2.4.** This way of representing axioms (and eventually theorems) is called the propositions as types paradigm.

SOME RIGHTS RESERVED

Asserting Truth in Mmt theories (continued)

- We can make `world1` happier by asserting *Mary loves Peter*.

```
    mlp = loves mary peter ❙
    mlp_axiom : ⊢mlp ❙
```

- Do *Peter and Mary love each other* in `world1`?
- We would have to have a proof of `plm ∧ mlp`, which we don't.
- **Observation:** There should be one, given that we have proofs for `plm` and `mlp`!
- **Observation:** We need a proof constructor – a function constant that constructs a proof of `plm ∧ mlp` from those.
- **Idea:** Let's just declare one: `pc : ⊢plm →⊢mlp →⊢plm ∧ mlp❙`
- We can generalize this to the inference rule of conjunction introduction

```
conjI : {A:o,B:o} ⊢A →⊢B →⊢A∧ B❙
```

`{A:o,B:o}` is the Mmt notation for Π from LF. (dependent type constructor)
Read as "for arbitrary but fixed propositions `A` and `B`..." ...

$$\frac{A \quad B}{A \wedge B}\mathcal{ND}\_0\wedge I$$

- **Idea:** This leads to a Mmt formalization of the propositional natural deduction calculus $\mathcal{ND}\_0$. (up next)

SOME RIGHTS RESERVED

Propositional Natural Deduction

- **Observation:** With the ideas discussed above we can do almost all of the inference rules of $\mathcal{ND}\_0$.

SOME RIGHTS RESERVED

Propositional Natural Deduction

- **Observation:** With the ideas discussed above we can do almost all of the inference rules of $\mathcal{ND}\_0$.
- **Let's start small** with $\Sigma_0 = \{\neg, \wedge\}$: here are the rules again.

Introduction

$$\frac{A \quad B}{A \wedge B}\mathcal{ND}\_0 \wedge I$$

$$\frac{\begin{array}{cc} [A]^1 & [A]^1 \\ \vdots & \vdots \\ C & \neg C \end{array}}{\neg A}\mathcal{ND}\_0 \neg I^1$$

Elimination

$$\frac{A \wedge B}{A}\mathcal{ND}\_0 \wedge E_l \qquad \frac{A \wedge B}{B}\mathcal{ND}\_0 \wedge E_r$$

$$\frac{\neg\neg A}{A}\mathcal{ND}\_0 \neg E$$

SOME RIGHTS RESERVED

Propositional Natural Deduction

- **Observation:** With the ideas discussed above we can do almost all of the inference rules of $\mathcal{ND}$_0.
- **Let's start small** with $\Sigma_0 = \{\neg, \wedge\}$: here are the rules again.
- The start of an Mmt theory:

```
theory proplog-ND : ur:?LF =
    include ?proplogMinimal
    ded : prop → type | # ⊢1
    conjI : {A:o,B:o} ⊢A → ⊢B → ⊢A∧ B
    conjEl : {A:o,B:o} ⊢A∧ B → ⊢A
    conjEr : {A:o,B:o} ⊢A∧ B → ⊢B
    negE : {A:o} ⊢¬¬A → ⊢A
```


SOME RIGHTS RESERVED

Local Hypotheses in Natural Deduction

▶ For $\mathcal{ND}\_0\neg I$ we need a new idea for the representation of the local hypothesis A.
A subproof P with a local hypothesis [A] allows to plug in a proof of A and complete it P to a full proof for C.
Idea: Represent this as a function from $\vdash$ A to $\vdash$ C.

$$\frac{\begin{array}{c}[A]^1\\ \vdots\\ C\end{array}\quad\begin{array}{c}[A]^1\\ \vdots\\ \neg C\end{array}}{\neg A}$$

▶ In Mmt we have:

```
   negI : {A:o,C:o} (⊢A →⊢C) →(⊢A →⊢¬C) →⊢¬A
```

$\mathcal{ND}\_0\neg I^1$ takes proof transformers as arguments and returns a proof of $\neg$A.

▶ With this idea, we can do the rest of the inference rules of $\mathcal{ND}\_0$, e.g.

```
implI: {a,b} (⊢a →⊢b) →⊢(a⇒ b)
```


SOME RIGHTS RESERVED

Writing Proofs in Mmt

- **Recap:** In Mmt, we can write axioms as declarations c : ⊢a using the propositions as types paradigm: the proof type ⊢a must be inhabited, since it has the proof c of a as an inhabitant.
- **Observation:** This can be extended to theorems, by giving denfinientia: A declaration c : ⊢a | = Φ also ensures that ⊢a is inhabited, but using already existing material Φ.
- **Example 2.5.** Let's try this on the well-known $\mathcal{ND}\_0$ proof

$$\dfrac{\dfrac{\dfrac{[A \wedge B]^1}{B}\mathcal{ND}\_0 \wedge E_r \quad \dfrac{[A \wedge B]^1}{A}\mathcal{ND}\_0 \wedge E_l}{B \wedge A}\mathcal{ND}\_0 \wedge I}{A \wedge B \Rightarrow B \wedge A}\mathcal{ND}\_0 \Rightarrow I^1$$

```
ac : {a,b} ⊢((a∧ b)⇒ (b∧ a)) |
   = [a, b] ([p:⊢(a∧ b)] (p andEr) (p andEl) andI) implI ▌
```


SOME RIGHTS RESERVED

Writing Proofs in Mmt (step by step)

- **Example 2.6 (Continued).**

$$\cfrac{\cfrac{\cfrac{[A \wedge B]^1}{B}\,\mathcal{ND}\_0 \wedge E_r \qquad \cfrac{[A \wedge B]^1}{A}\,\mathcal{ND}\_0 \wedge E_l}{B \wedge A}\,\mathcal{ND}\_0 \wedge I}{A \wedge B \Rightarrow B \wedge A}\,\mathcal{ND}\_0 \Rightarrow I^1$$

```
1 ac : {a,b} ⊢((a∧ b)⇒ (b∧ a))
2    = [a, b] ([p:⊢(a∧ b)]
3                (p andEr)
4                (p andEl)
5              andI)
6           implI
```

  - Line 1: name and type (optional)

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG
SOME RIGHTS RESERVED

Writing Proofs in Mmt (step by step)

- **Example 2.7 (Continued).**

$$\dfrac{\dfrac{\dfrac{[A \wedge B]^1}{B}\,\mathcal{ND}\_0 \wedge E_r \qquad \dfrac{[A \wedge B]^1}{A}\,\mathcal{ND}\_0}{B \wedge A}\,\mathcal{ND}\_0 \wedge}{A \wedge B \Rightarrow B \wedge A}\,\mathcal{ND}\_0 \Rightarrow I^1$$

```
ac : {a,b} ⊢((a∧ b)⇒ (b∧ a))    1
   = [a, b] ([p:⊢(a∧ b)]         2
                (p andEr)         3
                (p andEl)         4
              andI)               5
           implI                  6
```

  - Line 1: name and type (optional)
  - Line 2: λ-abstraction [a,b] corresponding to Π-abstraction {a,b}

SOME RIGHTS RESERVED

Writing Proofs in Mmt (step by step)

- **Example 2.8 (Continued).**

$$\dfrac{\dfrac{\dfrac{[A\wedge B]^1}{B}\,\mathcal{ND}\_0\wedge E_r \qquad \dfrac{[A\wedge B]^1}{A}\,\mathcal{ND}\_0\wedge}{B\wedge A}\,\mathcal{ND}\_0\wedge}{A\wedge B\Rightarrow B\wedge A}\,\mathcal{ND}\_0\Rightarrow I^1$$

```
ac : {a,b} ⊢((a∧ b)⇒ (b∧ a))1
   = [a, b] ([p:⊢(a∧ b)]      2
            (p andEr)         3
            (p andEl)         4
           andI)              5
        implI                 6
```

  - Line 1: name and type (optional)
  - Line 2: λ-abstraction [a,b] corresponding to Π-abstraction {a,b}
  - Line 6: the proof is constructed by impI with one argument (a subproof Ψ)

SOME RIGHTS RESERVED

Writing Proofs in Mmt (step by step)

- **Example 2.9 (Continued).**

$$\dfrac{\dfrac{\dfrac{[A \wedge B]^1}{B}\,\mathcal{ND}\_0 \wedge E_r \qquad \dfrac{[A \wedge B]^1}{A}\,\mathcal{ND}\_0}{B \wedge A}\,\mathcal{ND}\_0 \wedge}{A \wedge B \Rightarrow B \wedge A}\,\mathcal{ND}\_0 \Rightarrow I^1$$

```
ac : {a,b} ⊢((a∧ b)⇒ (b∧ a))     1
   = [a, b] ([p:⊢(a∧ b)]          2
                (p andEr)          3
                (p andEl)          4
                andI)              5
            implI                  6
```

  - Line 1: name and type (optional)
  - Line 2: λ-abstraction `[a,b]` corresponding to Π-abstraction `{a,b}`
  - Line 6: the proof is constructed by `impI` with one argument (a subproof Ψ)
    - **But remember**: `implI: {a,b} (⊢a →⊢b) →⊢(a⇒ b)` takes three!

Writing Proofs in Mmt (step by step)

- **Example 2.10 (Continued).**

$$\dfrac{\dfrac{\dfrac{[A \wedge B]^1}{B}\,\mathcal{ND}\_0\wedge E_r \qquad \dfrac{[A \wedge B]^1}{A}\,\mathcal{ND}\_0\wedge E_l}{B \wedge A}\,\mathcal{ND}\_0\wedge I}{A \wedge B \Rightarrow B \wedge A}\,\mathcal{ND}\_0 \Rightarrow I^1$$

```
ac : {a,b} ⊢((a∧ b)⇒ (b∧ a))   1
   = [a, b] ([p:⊢(a∧ b)]       2
              (p andEr)         3
              (p andEl)         4
             andI)              5
          implI                 6
```

  - Line 1: name and type (optional)
  - Line 2: λ-abstraction [a,b] corresponding to Π-abstraction {a,b}
  - Line 6: the proof is constructed by impI with one argument (a subproof Ψ)
    - **But remember**: implI: {a,b} (⊢a →⊢b) →⊢(a⇒ b) takes three!
    - **Idea**: add special postfix notation definition # 3 impI ($3 \mapsto \Psi$)

SOME RIGHTS RESERVED

Writing Proofs in Mmt (step by step)

- **Example 2.11 (Continued).**

$$\dfrac{\dfrac{\dfrac{[A\wedge B]^1}{B}\,\mathcal{ND}\_0\wedge E_r \qquad \dfrac{[A\wedge B]^1}{A}\,\mathcal{ND}\_0\wedge E_l}{B\wedge A}\,\mathcal{ND}\_0\wedge I}{A\wedge B\Rightarrow B\wedge A}\,\mathcal{ND}\_0\Rightarrow I^1$$

```
ac : {a,b} ⊢((a∧ b)⇒ (b∧ a))    1
   = [a, b] ([p:⊢(a∧ b)]        2
              (p andEr)         3
              (p andEl)         4
             andI)              5
          implI                 6
```

  - Line 1: name and type (optional)
  - Line 2: λ-abstraction `[a,b]` corresponding to Π-abstraction `{a,b}`
  - Line 6: the proof is constructed by `impI` with one argument (a subproof Ψ)
    - **But remember**: `implI: {a,b} (⊢a →⊢b) →⊢(a⇒ b)` takes three!
    - **Idea**: add special postfix notation definition `# 3 impI` ($3 \mapsto \Psi$)
    - **Justification**: The Mmt system can reconstruct implicit arguments

SOME RIGHTS RESERVED

Writing Proofs in Mmt (step by step)

- **Example 2.12 (Continued).**

$$\dfrac{\dfrac{\dfrac{[A\wedge B]^1}{B}\,\mathcal{ND}\_0\wedge E_r \qquad \dfrac{[A\wedge B]^1}{A}\,\mathcal{ND}\_0}{B\wedge A}\,\mathcal{ND}\_0\wedge}{A\wedge B\Rightarrow B\wedge A}\,\mathcal{ND}\_0\Rightarrow I^1$$

```
ac : {a,b} ⊢((a∧ b)⇒ (b∧ a))1
   = [a, b] ([p:⊢(a∧ b)]      2
                (p andEr)     3
                (p andEl)     4
               andI)          5
            implI             6
```

- Line 1: name and type (optional)
- Line 2: λ-abstraction `[a,b]` corresponding to Π-abstraction `{a,b}`
- Line 6: the proof is constructed by `impI` with one argument (a subproof Ψ)
  - **But remember**: `implI: {a,b} (⊢a →⊢b) →⊢(a⇒ b)` takes three!
  - **Idea**: add special postfix notation definition `# 3 impI` ($3 \mapsto \Psi$)
  - **Justification**: The Mmt system can reconstruct implicit arguments
- Lines 2-5: Subproof Ψ with local hyp. $[a\wedge b]^1$, represented as λp-term in Line 4
  Idea: the (informal) function of the co-indexing is formalized by λ-abstraction

SOME RIGHTS RESERVED

- **Example 2.13 (Continued).**

$$\dfrac{\dfrac{\dfrac{[A \wedge B]^1}{B}\mathcal{ND}\_0 \wedge E_r \qquad \dfrac{[A \wedge B]^1}{A}\mathcal{ND}\_0}{B \wedge A}\mathcal{ND}\_0 \wedge}{A \wedge B \Rightarrow B \wedge A}\mathcal{ND}\_0 \Rightarrow I^1$$

```
ac : {a,b} ⊢((a∧ b)⇒ (b∧ a))     1
   = [a, b] ([p:⊢(a∧ b)]         2
              (p andEr)          3
              (p andEl)          4
             andI)               5
           implI                 6
```

  - Line 1: name and type (optional)
  - Line 2: λ-abstraction `[a,b]` corresponding to Π-abstraction `{a,b}`
  - Line 6: the proof is constructed by `impI` with one argument (a subproof Ψ)
    - **But remember**: `implI: {a,b} (⊢a →⊢b) →⊢(a⇒ b)` takes three!
    - **Idea**: add special postfix notation definition `# 3 impI` $(3 \mapsto \Psi)$
    - **Justification**: The Mmt system can reconstruct implicit arguments
  - Lines 2-5: Subproof Ψ with local hyp. $[a \wedge b]^1$, represented as λp-term in Line 4
    Idea: the (informal) function of the co-indexing is formalized by λ-abstraction
  - Line 5: result of Ψ constructed by `andI` – notation definition `# 3 4 andI`

SOME RIGHTS RESERVED

Writing Proofs in Mmt (step by step)

- **Example 2.14 (Continued).**

$$\cfrac{\cfrac{\cfrac{[A\wedge B]^1}{B}\;\mathcal{ND}\_0\wedge E_r \qquad \cfrac{[A\wedge B]^1}{A}\;\mathcal{ND}\_0\wedge}{B\wedge A}\;\mathcal{ND}\_0\wedge}{A\wedge B\Rightarrow B\wedge A}\;\mathcal{ND}\_0\Rightarrow I^1$$

```
ac : {a,b} ⊢((a∧ b)⇒ (b∧ a))   1
 = [a, b] ([p:⊢(a∧ b)]          2
            (p andEr)           3
            (p andEl)           4
            andI)               5
          implI                 6
```

- Line 1: name and type (optional)
- Line 2: λ-abstraction `[a,b]` corresponding to Π-abstraction `{a,b}`
- Line 6: the proof is constructed by `impI` with one argument (a subproof Ψ)
  - **But remember**: `implI: {a,b} (⊢a →⊢b) →⊢(a⇒ b)` takes three!
  - **Idea**: add special postfix notation definition `# 3 impI` ($3\mapsto\Psi$)
  - **Justification**: The Mmt system can reconstruct implicit arguments
- Lines 2-5: Subproof Ψ with local hyp. $[a\wedge b]^1$, represented as λp-term in Line 4
  Idea: the (informal) function of the co-indexing is formalized by λ-abstraction
- Line 5: result of Ψ constructed by `andI` – notation definition `# 3 4 andI`
- Line 3/4: two subproofs constructed from p by `andEl`/`andEr`.

SOME RIGHTS RESERVED

Writing Proofs in Mmt (step by step)

- **Example 2.15 (Continued).**

$$\dfrac{\dfrac{\dfrac{[A\wedge B]^1}{B}\,\mathcal{ND}\_0\wedge E_r \qquad \dfrac{[A\wedge B]^1}{A}\,\mathcal{ND}\_0\wedge E}{B\wedge A}\,\mathcal{ND}\_0\wedge}{A\wedge B\Rightarrow B\wedge A}\,\mathcal{ND}\_0\Rightarrow I^1$$

```
ac : {a,b} ⊢((a∧ b)⇒ (b∧ a))   1
   = [a, b] ([p:⊢(a∧ b)]       2
                (p andEr)      3
                (p andEl)      4
                andI)          5
              implI            6
```

  - Line 1: name and type (optional)
  - Line 2: λ-abstraction `[a,b]` corresponding to Π-abstraction `{a,b}`
  - Line 6: the proof is constructed by `impI` with one argument (a subproof Ψ)
    - **But remember**: `implI: {a,b} (⊢a →⊢b) →⊢(a⇒ b)` takes three!
    - **Idea**: add special postfix notation definition `# 3 impI` $(3 \mapsto \Psi)$
    - **Justification**: The Mmt system can reconstruct implicit arguments
  - Lines 2-5: Subproof Ψ with local hyp. $[a\wedge b]^1$, represented as λp-term in Line 4
    Idea: the (informal) function of the co-indexing is formalized by λ-abstraction
  - Line 5: result of Ψ constructed by `andI` – notation definition `# 3 4 andI`
  - Line 3/4: two subproofs constructed from p by `andEl`/`andEr`.
- **Observation 1:** The postfix notations make the Mmt proof term similar!
- **Observation 2:** But writing them is very tedious and complex still.

SOME RIGHTS RESERVED

Modular Representation in Mmt

- **Recall:** We said that for PL^0, it does not matter if $\Sigma_0 = \{\neg, \wedge\}$ or $\Sigma_0 = \{\neg, \vee\}$.
- **In particular** we can always inter-define $\wedge$ and $\vee$ via de-Morgan.
- Let's make this formal using views.
- **Example 2.16.** A modular development of the two variants of PL^0

```
theory dednot : ur:?LF =
    prop : type | # o ❙
    ded : o → type | # ⊢1 ❙
    not : o → o | # ¬1 ❚
```

```
theory notand : ur:?LF =
    include ?dednot ❙
    and : o → o → o | # 1 ∧ 2 ❙
    andI : {a,b} ⊢a →⊢b →
⊢(a∧ b) ❚
```

```
theory notor : ur:?LF =
    include ?dednot ❙
    or : o → o → o | # 1 ∨ 2 ❙
    orIl : {a,b} ⊢a →
⊢(a∨ b) ❙
    orIr : {a,b} ⊢b →
⊢(a∨ b) ❚
```

```
view and2or : ?notand -> ?notor =
     and = [a,b] ¬((¬a) ∨
(¬b)) ❙
     andI = Φ ❚
```

```
view or2and : ?notor -> ?notan
     or = [a,b] ¬((¬a) ∧
(¬b)) ❙
     andI = Ψ ❚
```

For some suitable proof expressions Φ and Ψ.

SOME RIGHTS RESERVED

3.2.2 General Functionality of MMT

SOME RIGHTS RESERVED

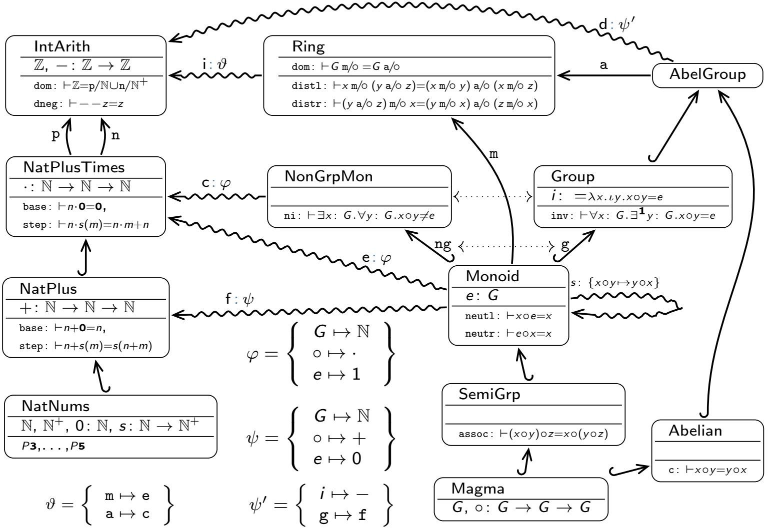
Representation language (Mmt)

- **Definition 2.17.** Mmt $\widehat{=}$ module system for mathematical theories
- Formal syntax and semantics
  - needed for mathematical interface language
  - but how to avoid foundational commitment?
- Foundation-independence
  - identify aspects of underlying language that are necessary for large scale processing
  - formalize exactly those, be parametric in the rest
  - observation: most large scale operations need the same aspects
- Module system
  - preserve mathematical structure wherever possible
  - formal semantics for modularity
- Web-scalable
  - build on XML, OpenMath, OMDoc
  - URI based logical identifiers for all declarations
- Implemented in the Mmt system.

SOME RIGHTS RESERVED

Modular Representation of Math (Mmt Example)

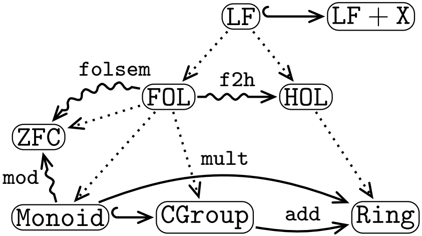
- **Example 2.18 (Elementary Algebra and Arithmetics).**



SOME RIGHTS RESERVED

Representing Logics and Foundations as Theories

- **Example 2.19.** Logics and foundations represented as Mmt theories



- **Definition 2.20.** Meta relation between theories special case of inclusion
- **Uniform Meaning Space:** morphisms between formalizations in different logics become possible via meta-morphisms.
- *Remark 2.21.* Semantics of logics as views into foundations, e.g., `folsem`.
- *Remark 2.22.* Models represented as views into foundations (e.g. ZFC)
- **Example 2.23.** $\mathtt{mod} := \{G \mapsto \mathbb{Z}, \circ \mapsto +, e \mapsto 0\}$ interprets Monoid in ZFC.

SOME RIGHTS RESERVED

A MitM Theory in Mmt Surface Language

- **Example 2.24.** A theory of Groups

Declaration $\widehat{=}$
name : type [= Def] [# notation]

Axioms $\widehat{=}$ Declaration with type $\vdash F$

ModelsOf makes a record type from a theory.

```
theory group : base:?Logic =
  theory group_theory : base:?Logic =
    include ?monoid/monoid_theory

    inverse : U → U  # 1 ⁻¹  prec 24
    inverseproperty : ⊦ ∀ [x] x ∘ x ⁻¹ ≐ e

  group = ModelsOf group_theory
```

- **MitM Foundation:** optimized for natural math formulation
  - higher-order logic based on polymorphic λ-calculus
  - judgments-as-types paradigm: $\vdash F \,\widehat{=}$ type of proofs of F
  - dependent types with predicate subtyping, e.g. $\{n\}\{'a \in \mathrm{mat}(n,n) | \mathrm{symm}(a)'\}$
  - (dependent) record types for reflecting theories

SOME RIGHTS RESERVED

The Mmt Module System

- **Central notion:** Theory graph with theory nodes and theory morphisms as edges.
- **Definition 2.25.** In Mmt, a theory is a sequence of constant declarations optionally with type declarations and definitions.
- Mmt employs the Curry/Howard isomorphism and treats
  - axioms/conjectures as typed symbol declarations (propositions-as-types)
  - inference rules as function types (proof transformers)
  - theorems as definitions (proof terms for conjectures)
- **Definition 2.26.** Mmt has two kinds of theory morphisms
  - structures instantiate theories in a new context (also called: definitional link, import) they import theory S into theory T (induces theory morphism $S \to T$)
  - views translate between existing theories (also called: postulated link, theorem link) Views transport theorem from source to target (framing).
- Together, structures and views allow a very high degree of re-use
- **Definition 2.27.** We call a statement t induced in a theory T, iff there is
  - a path of theory morphisms from a theory S to T with (joint) assignment σ,
  - such that $t = \sigma(s)$ for some statement s in S.
- **Definition 2.28.** In Mmt, all induced statements have a canonical name, the MMT URI.

SOME RIGHTS RESERVED

3.3 ELPI a Higher-Order Logic Programming Language

SOME RIGHTS RESERVED

ELPI

- **Definition 3.1.** λProlog, also written lambda Prolog, is a logic programming language featuring polymorphic typing, modular programming, and h i g h e r - o r d e r f u n c t i o nhigher-order programming.
- **Definition 3.2.** ELPI implements a variant of λProlog enriched with constraint handling rules.

SOME RIGHTS RESERVED

ELPI by example

- **Intuition:** ELPI almost works like Prolog, if we forget the advanced features
- **But:** ELPI insists on types declarations for all objects it works with.
- **Example 3.3 (A Member Predicate).** Indeed in line 1 we see an ELPI type declaration for the `ismember` predicate. As in Prolog, we use identifiers starting with capital letters for variables. This makes `ismember` polymorphic in the type T.

```
1 type ismember T -> list T -> prop.
2 ismember X [X|_T].
3 ismember X [_H|T] :- ismember X T.
```

  The recursive `ismember` predicate itself is just as we would write it in Prolog. As always, we can test this with the queries
  - `ismember 2 [1,2,3]` which succeeds and
  - `ismember 5 [1,2,3]` which fails.

SOME RIGHTS RESERVED

Propositional Logic in ELPI

- **Remember:** we wanted to use ELPI to automate proof construction for our target logics.
- **Idea:** Let's just start with PL^0 – this is really just like in Mmt.

```
kind oo type. % propositions (prop and o are taken)
type neg oo -> oo.
type and oo -> oo -> oo.
type or oo -> oo -> oo.
type impl oo -> oo -> oo.
type true oo.
type false oo.
type pvar int -> oo.
```

The declarations (and their ELPI syntax) should be quite obvious
the `pvar` function makes a countable collection of propositional variables.

SOME RIGHTS RESERVED

Predicates for Properties of Formulae

- **Problem:** We will need to know when a PL^0 formula is atomic later.
- **Idea:** It is easier to (first) specify whehter a formula is complex.

```
type complex oo -> prop.
complex (neg _Y).
complex (and _X _Y).
```

  And then we just make atomic to be "not complex".
- **Standard Method:** In ELPI, we use negation as failure: To establish that a term t is atomic we try to establish that it complex and if that succeeds, then we fail.
  On the other hand, if the first clause of the `atomic` predicate fails, then the second clause (automatically) succeeeds.
  Together they switch orchestrate the switch of truth values needed for negation as failure

```
type atomic oo -> prop.
atomic (X) :- complex(X),!,fail.
atomic (_X).
```

  The trick now is to guard the `fail` with a cut operator `!`, a literal that forbids the `atomic` predicate to backtrack after it failed. Otherwise the first clause would succeed via the second clause ruining the effect.

SOME RIGHTS RESERVED

Part 1
English as a Formal Language: The Method of Fragments

SOME RIGHTS RESERVED

Chapter 4
Logic as a Tool for Modeling NL Semantics

SOME RIGHTS RESERVED

4.1 The Method of Fragments

SOME RIGHTS RESERVED

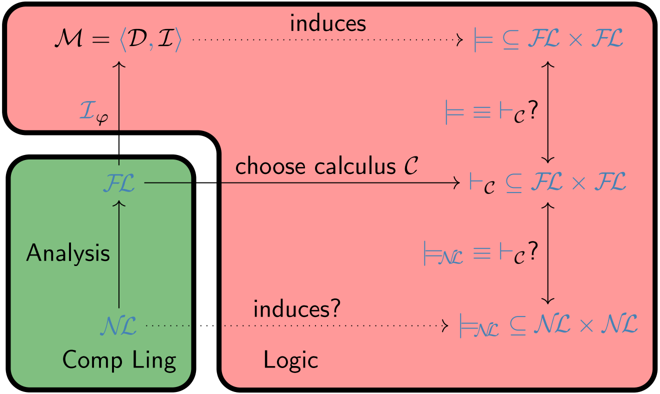
Natural Language Fragments

- **Methodological Problem:** How to organize the scientific method for natural language?
- **Delineation Problem:** What is natural language, e.g. English? Which Aspects do we want to study?
- **Idea:** Formalize a set (NL) sentences we want to study by a grammar $\leadsto$ Richard Montague's method of fragments (1972).
- **Definition 1.1.** The language L of a context-free grammar is called a fragment of a natural language N, iff $L \subseteq N$.
- **Scientific Fiction:** We can exhaust English with ever-increasing fragments, develop a semantic meaning theory for each.
- **Idea:** Use nonterminals to classify NL phrases.
- **Definition 1.2.** We call a nonterminal symbol of a context-free grammar a phrasal category. We distinguish two kinds of rules:
  - structural rules: $\mathcal{L}\colon H{\rightarrow}c_1, \ldots, c_n$ with head H, label $\mathcal{L}$, and a sequence of phrasal categories c_i.
  - lexical rules: $\mathcal{L}\colon H{\rightarrow}t_1 \mid \ldots \mid t_n$, where the t_i are terminals (i.e. NL phrases)
- **Definition 1.3.** In the method of fragments we use a CFG to parse sentences from the fragment into an abstract syntax tree (AST) for further processing.

SOME RIGHTS RESERVED

Formal Natural Language Semantics with Fragments

- **Idea:** We will follow the picture we have discussed before



Choose a target logic $\mathcal{FL}$ and specify a translation from syntax trees to formulae!

SOME RIGHTS RESERVED

Semantics by Translation

- **Idea:** We translate sentences by translating their syntax trees via tree node translation rules.
- **Note:** This makes the induced meaning theory compositional.
- **Definition 1.4.** We represent a node α in a syntax tree with children $\beta_1, \ldots, \beta_n$ by $[X_{1\beta_1}, \ldots, X_{n\beta_n}]_\alpha$ and write a translation rule as

$$\mathcal{L}\colon [X_{1\beta_1}, \ldots, X_{n\beta_n}]_\alpha \leadsto \Phi({X_1}', \ldots, {X_n}')$$

  if the translation of the node α can be computed from those of the β_i via a semantical function Φ.
- **Definition 1.5.** For a natural language utterance A, we will use $\langle A \rangle$ for the result of translating A.
- **Definition 1.6 (Default Rule).** For every word w in the fragment we assume a constant w' in the logic $\mathcal{L}$ and the "pseudo-rule" $t1\colon w \leadsto w'$. (if no other translation rule applies)

SOME RIGHTS RESERVED

4.2 What is Logic?

SOME RIGHTS RESERVED

What is Logic?

- **Definition 2.1.** Logic $\widehat{=}$ formal languages, inference and their relation with the world
  - Formal language $\mathcal{FL}$: set of formulae ($2+3/7$, $\forall x.x+y=y+x$)
  - Formula: sequence/tree of symbols ($x, y, f, g, p, 1, \pi, \in, \neg, \forall, \exists$)
  - Model: things we understand (e.g. number theory)
  - Interpretation: maps formulae into models ($[\![\text{three plus five}]\!]^{\mathcal{I}} = 8$)
  - Validity: $\mathcal{M} \models \mathrm{A}$, iff $[\![\mathrm{A}]\!]^{\mathcal{I}} = \mathsf{T}$ (five greater three is valid)
  - Entailment: $\mathrm{A} \models \mathrm{B}$, iff $\mathcal{M} \models \mathrm{B}$ for all $\mathcal{M} \models \mathrm{A}$. (generalize to $\mathcal{H} \models \mathrm{A}$)
  - Inference: rules to transform (sets of) formulae ($\mathrm{A}, \mathrm{A} \Rightarrow \mathrm{B} \vdash \mathrm{B}$)
  - Syntax: formulae, inference (just a bunch of symbols)
  - Semantics: models, interpr., validity, entailment (math. structures)
- **Important Question:** relation between syntax and semantics?

SOME RIGHTS RESERVED

4.3 Using Logic to Model Meaning of Natural Language

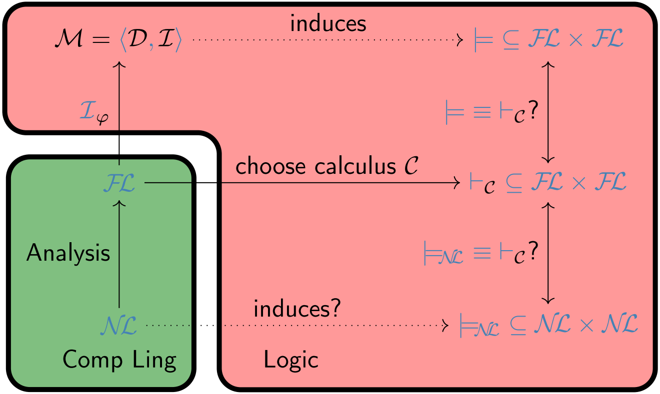
SOME RIGHTS RESERVED

Modeling Natural Language Semantics

- **Problem:** Find formal (logic) system for the meaning of natural language.
- History of ideas
  - Propositional logic [ancient Greeks like Aristotle]
    * *Every human is mortal*
  - First-Order Predicate logic [Frege $\leq$ 1900]
    * *I believe, that my audience already knows this.*
  - Modal logic [Lewis18, Kripke65]
    * *A man sleeps. He snores.* $((\exists X.\mathrm{man}(X) \wedge \mathrm{sleeps}(X))) \wedge \mathrm{snores}(X)$
  - Various dynamic approaches (e.g. DRT, DPL)
    * *Most men wear black*
  - Higher-order Logic, e.g. generalized quantifiers
  - ...

SOME RIGHTS RESERVED

Natural Language Semantics?



SOME RIGHTS RESERVED

Logic-Based Knowledge Representation for NLP

- Logic (and related formalisms) allow to integrate world knowledge
  - explicitly (gives more understanding than statistical methods)
  - transparently (symbolic methods are monotonic)
  - systematically (we can prove theorems about our systems)
- Signal + World knowledge makes more powerful model
  - Does not preclude the use of statistical methods to guide inference
- Problems with logic-based approaches
  - Where does the world knowledge come from? (Ontology problem)
  - How to guide search induced by log. calculi (combinatorial explosion)

SOME RIGHTS RESERVED

Chapter 5
Fragment 1

SOME RIGHTS RESERVED

5.1 The First Fragment: Setting up the Basics

SOME RIGHTS RESERVED

Fragment 1 Data (Sentences we want to cover)

- **Fragment 1 Data:** We delineate the intended fragment by giving examples
  1. *Ethel kicked the cat and Fiona laughted*
  2. *Peter is the teacher*
  3. *The teacher is happy*
  4. *It is not the case that Bertie ran*
  5. *It is not the case that Jo is happy*
- We can later use these sentences as benchmark tests.

SOME RIGHTS RESERVED

5.1.1 Natural Language Syntax (Fragment 1)

SOME RIGHTS RESERVED

Structural Grammar Rules

- **Definition 1.1.** $\mathcal{F}_1$ knows the following eight phrasal categories

S	sentence	NP	noun phrase
N	noun	N_{pr}	proper name
V^i	intransitive verb	V^t	transitive verb
conj	connective	Adj	adjective

- **Definition 1.2.** We have the following production rules in $\mathcal{F}_1$.

  $S1$: $S \rightarrow \mathrm{NP}, V^i$,
  $S2$: $S \rightarrow \mathrm{NP}, V^t, \mathrm{NP}$,
  $N1$: $\mathrm{NP} \rightarrow N_{\mathrm{pr}}$,
  $N2$: $\mathrm{NP} \rightarrow$ the, N,
  $S3$: $S \rightarrow$ It is not the case that, S,
  $S4$: $S \rightarrow S$, conj, S,
  $S5$: $S \rightarrow \mathrm{NP}$, is, NP,
  $S6$: $S \rightarrow \mathrm{NP}$, is, Adj

SOME RIGHTS RESERVED

Lexical insertion rules for Fragment 1

- **Definition 1.3.** We have the following lexical rules in Fragment 1.

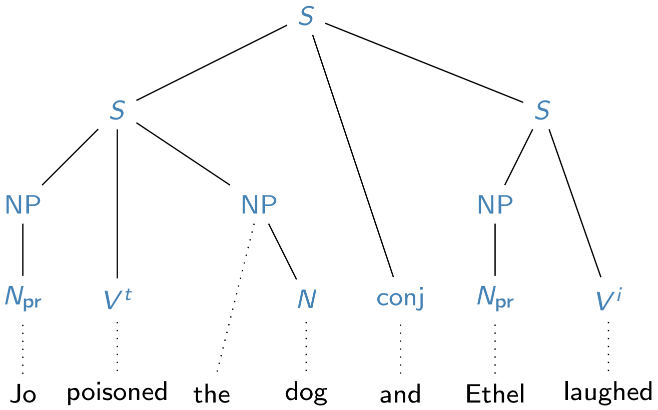
$$L1\colon N_{\mathrm{pr}} \rightarrow \mathrm{Prudence} \mid \mathrm{Ethel} \mid \mathrm{Chester} \mid \mathrm{Jo} \mid \mathrm{Bertie} \mid \mathrm{Fiona},$$
$$L2\colon N \rightarrow \mathrm{book} \mid \mathrm{cake} \mid \mathrm{cat} \mid \mathrm{golfer} \mid \mathrm{dog} \mid \mathrm{lecturer} \mid \mathrm{student} \mid \mathrm{singer},$$
$$L3\colon V^i \rightarrow \mathrm{ran} \mid \mathrm{laughed} \mid \mathrm{sang} \mid \mathrm{howled} \mid \mathrm{screamed},$$
$$L4\colon V^t \rightarrow \mathrm{read} \mid \mathrm{poisoned} \mid \mathrm{ate} \mid \mathrm{liked} \mid \mathrm{loathed} \mid \mathrm{kicked},\ L5\colon \mathrm{conj} \rightarrow \mathrm{and} \mid \mathrm{or},$$
$$L6\colon \mathrm{Adj} \rightarrow \mathrm{happy} \mid \mathrm{crazy} \mid \mathrm{messy} \mid \mathrm{disgusting} \mid \mathrm{wealthy}$$

- Note: We will adopt the convention that new lexical rules can be generated spontaneously as needed.

SOME RIGHTS RESERVED

Syntax Example: *Jo poisoned the dog and Ethel laughed*

- **Observation 1.4.** *Jo poisoned the dog and Ethel laughed is a sentence of fragment 1*
- We can construct a syntax tree for it!



SOME RIGHTS RESERVED

5.1.2 Predicate Logic without Quantifiers

SOME RIGHTS RESERVED

Individuals and their Properties/Relations

- **Observation:** We want to talk about individuals like Stefan, Nicole, and Jochen and their properties, e.g. being blond, or studying AI and relationships, e.g. that *Stefan loves Nicole*.
- **Idea:** Re-use PL^0, but replace propositional variables with something more expressive! (instead of fancy variable name trick)

SOME RIGHTS RESERVED

Individuals and their Properties/Relations

- **Observation:** We want to talk about individuals like Stefan, Nicole, and Jochen and their properties, e.g. being blond, or studying AI and relationships, e.g. that *Stefan loves Nicole*.
- **Idea:** Re-use PL^0, but replace propositional variables with something more expressive! (instead of fancy variable name trick)
- **Definition 1.7.** A first-order signature $\langle\Sigma^f, \Sigma^p\rangle$ consists of
  - $\Sigma^f := \bigcup_{k\in\mathbb{N}} \Sigma_k^f$ of function constants, where members of Σ_k^f denote k-ary functions on individuals,
  - $\Sigma^p := \bigcup_{k\in\mathbb{N}} \Sigma_k^p$ of predicate constants, where members of Σ_k^p denote k-ary relations among individuals,

  where Σ_k^f and Σ_k^p are pairwise disjoint, countable sets of symbols for each $k\in\mathbb{N}$.

SOME RIGHTS RESERVED

Individuals and their Properties/Relations

- **Observation:** We want to talk about individuals like Stefan, Nicole, and Jochen and their properties, e.g. being blond, or studying AI and relationships, e.g. that *Stefan loves Nicole*.
- **Idea:** Re-use PL^0, but replace propositional variables with something more expressive! (instead of fancy variable name trick)
- **Definition 1.9.** A first-order signature $\langle\Sigma^f,\Sigma^p\rangle$ consists of
  - $\Sigma^f:=\bigcup_{k\in\mathbb{N}}\Sigma^f_k$ of function constants, where members of Σ^f_k denote k-ary functions on individuals,
  - $\Sigma^p:=\bigcup_{k\in\mathbb{N}}\Sigma^p_k$ of predicate constants, where members of Σ^p_k denote k-ary relations among individuals,

  where Σ^f_k and Σ^p_k are pairwise disjoint, countable sets of symbols for each $k\in\mathbb{N}$.
- **Definition 1.10.** The formulae of PL^{nq} are given by the following grammar

function constants	f^k	$\in$	Σ^f_k	
predicate constants	p^k	$\in$	Σ^p_k	
terms	t	$::=$	f^0	constant
		$\mid$	$f^k(t_1,\ldots,t_k)$	application
formulae	A	$::=$	$p^k(t_1,\ldots,t_k)$	atomic
		$\mid$	$\neg A$	negation
		$\mid$	$A_1\wedge A_2$	conjunction

SOME RIGHTS RESERVED

PL^{nq} Semantics

- **Definition 1.11.** Domains $\mathcal{D}_0 = \{T, F\}$ of truth values and $\mathcal{D}_\iota \neq \emptyset$ of individuals.
- **Definition 1.12.** Interpretation $\mathcal{I}$ assigns values to constants, e.g.
  - $\mathcal{I}(\neg)\colon \mathcal{D}_0 \to \mathcal{D}_0; T \mapsto F; F \mapsto T$ and $\mathcal{I}(\wedge) = \ldots$ (as in PL^0)
  - $\mathcal{I}\colon \Sigma_0^f \to \mathcal{D}_\iota$ (interpret individual constants as individuals)
  - $\mathcal{I}\colon \Sigma_k^f \to {\mathcal{D}_\iota}^k \to \mathcal{D}_\iota$ (interpret function constants as functions)
  - $\mathcal{I}\colon \Sigma_k^p \to \mathcal{P}({\mathcal{D}_\iota}^k)$ (interpret predicate constants as relations)
- **Definition 1.13.** The value function $\mathcal{I}$ assigns values to formulae: (recursively)
  - $\mathcal{I}(f(A^1, \ldots, A^k)) := \mathcal{I}(f)(\mathcal{I}(A^1), \ldots, \mathcal{I}(A^k))$
  - $\mathcal{I}(p(A^1, \ldots, A^k)) := T$, iff $\langle \mathcal{I}(A^1), \ldots, \mathcal{I}(A^k) \rangle \in \mathcal{I}(p)$
  - $\mathcal{I}(\neg A) = \mathcal{I}(\neg)(\mathcal{I}(A))$ and $\mathcal{I}(A \wedge B) = \mathcal{I}(\wedge)(\mathcal{I}(A), \mathcal{I}(G))$ (just as in PL^0)
- **Definition 1.14.** Model: $\mathcal{M} = \langle \mathcal{D}_\iota, \mathcal{I} \rangle$ varies in $\mathcal{D}_\iota$ and $\mathcal{I}$.
- **Theorem 1.15.** *PL^{nq} is isomorphic to PL^0* (*interpret atoms as prop. variables*)

SOME RIGHTS RESERVED

- **Example 1.16.** Let $L := \{a, b, c, d, e, P, Q, R, S\}$, we set the universe $\mathcal{D} := \{\clubsuit, \spadesuit, \heartsuit, \diamondsuit\}$, and specify the interpretation function $\mathcal{I}$ by setting
  - $a \mapsto \clubsuit$, $b \mapsto \spadesuit$, $c \mapsto \heartsuit$, $d \mapsto \diamondsuit$, and $e \mapsto \diamondsuit$ for constants,
  - $P \mapsto \{\clubsuit, \spadesuit\}$ and $Q \mapsto \{\spadesuit, \diamondsuit\}$, for unary predicate constants.
  - $R \mapsto \{\langle \heartsuit, \diamondsuit \rangle, \langle \diamondsuit, \heartsuit \rangle\}$, and $S \mapsto \{\langle \diamondsuit, \spadesuit \rangle, \langle \spadesuit, \clubsuit \rangle\}$ for binary predicate constants.
- **Example 1.17 (Computing Meaning in this Model).**
  - $\mathcal{I}(R(a, b) \wedge P(c)) = \mathsf{T}$, iff
  - $\mathcal{I}(R(a, b)) = \mathsf{T}$ and $\mathcal{I}(P(c)) = \mathsf{T}$, iff
  - $\langle \mathcal{I}(a), \mathcal{I}(b) \rangle \in \mathcal{I}(R)$ and $\mathcal{I}(c) \in \mathcal{I}(P)$, iff
  - $\langle \clubsuit, \spadesuit \rangle \in \{\langle \heartsuit, \diamondsuit \rangle, \langle \diamondsuit, \heartsuit \rangle\}$ and $\heartsuit \in \{\clubsuit, \spadesuit\}$

  So, $\mathcal{I}(R(a, b) \wedge P(c)) = \mathsf{F}$.

SOME RIGHTS RESERVED

PL^{nq} and PL^0 are Isomorphic

- **Observation:** For every choice of Σ of signature, the set $\mathcal{A}_\Sigma$ of atomic PL^{nq} formulae is countable, so there is a $\mathcal{V}_\Sigma \subseteq \mathcal{V}_0$ and a bijection $\theta_\Sigma \colon \mathcal{A}_\Sigma \to \mathcal{V}_\Sigma$. θ_Σ can be extended to formulae as PL^{nq} and PL^0 share connectives.
- **Lemma 1.18.** *For every model $\mathcal{M} = \langle \mathcal{D}_\iota, \mathcal{I} \rangle$, there is a variable assignment $\varphi_\mathcal{M}$, such that $\mathcal{I}_{\varphi_\mathcal{M}}(\mathrm{A}) = \mathcal{I}(\mathrm{A})$.*
- *Proof sketch:* We just define $\varphi_\mathcal{M}(X) := \mathcal{I}(\theta_\Sigma^{-1}(X))$
- **Lemma 1.19.** *For every variable assignment $\psi \colon \mathcal{V}_\Sigma \to \{\mathsf{T}, \mathsf{F}\}$ there is a model $\mathcal{M}^\psi = \langle \mathcal{D}^\psi, \mathcal{I}^\psi \rangle$, such that $\mathcal{I}_\psi(\mathrm{A}) = \mathcal{I}^\psi(\mathrm{A})$.*
- *Proof sketch:* see next slide
- **Corollary 1.20.** *PL^{nq} is isomorphic to PL^0, i.e. the following diagram commutes:*

$$\begin{tikzcd}
\langle \mathcal{D}^\psi, \mathcal{I}^\psi \rangle & \mathcal{V}_\Sigma \to \{\mathsf{T}, \mathsf{F}\} \arrow[l, "\psi \mapsto \mathcal{M}^\psi"'] \\
PL^{nq}(\Sigma) \arrow[u, "\mathcal{I}^\psi()"] \arrow[r, "\theta_\Sigma"] & PL^0(\mathcal{A}_\Sigma) \arrow[u, "\mathcal{I}_{\varphi_\mathcal{M}}()"']
\end{tikzcd}$$

- **Note:** This constellation with a language isomorphism and a corresponding model isomorphism (in converse direction) is typical for a logic isomorphism.

SOME RIGHTS RESERVED

Valuation and Satisfiability

- **Lemma 1.21.** *For every variable assignment $\psi\colon \mathcal{V}_\Sigma \to \{\mathsf{T},\mathsf{F}\}$ there is a model $\mathcal{M}^\psi = \langle \mathcal{D}^\psi, \mathcal{I}^\psi \rangle$, such that $\mathcal{I}_\psi(\mathrm{A}) = \mathcal{I}^\psi(\mathrm{A})$.*
- *Proof:* We construct $\mathcal{M}^\psi = \langle \mathcal{D}^\psi, \mathcal{I}^\psi \rangle$ and show that it works as desired.
  1. Let $\mathcal{D}^\psi$ be the set of $\mathrm{PL^{nq}}$ terms over Σ, and
     - $\mathcal{I}^\psi(f)\colon {\mathcal{D}_\iota}^k \to {\mathcal{D}^\psi}^k; \langle \mathrm{A}_1, \ldots, \mathrm{A}_k \rangle \mapsto f(\mathrm{A}_1, \ldots, \mathrm{A}_k)$ for $f \in \Sigma^f_k$
     - $\mathcal{I}^\psi(p) := \{\langle \mathrm{A}_1, \ldots, \mathrm{A}_k \rangle \mid \psi(\theta_\psi^{-1} p(\mathrm{A}_1, \ldots, \mathrm{A}_k)) = \mathsf{T}\}$ for $p \in \Sigma^p$.
  2. We show $\mathcal{I}^\psi(\mathrm{A}) = \mathrm{A}$ for terms A by induction on A
     1. If $\mathrm{A} = c$, then $\mathcal{I}^\psi(\mathrm{A}) = \mathcal{I}^\psi(c) = c = \mathrm{A}$
     2. If $\mathrm{A} = f(\mathrm{A}_1, \ldots, \mathrm{A}_n)$ then
        $\mathcal{I}^\psi(\mathrm{A}) = \mathcal{I}^\psi(f)(\mathcal{I}(\mathrm{A}_1), \ldots, \mathcal{I}(\mathrm{A}_n)) = \mathcal{I}^\psi(f)(\mathrm{A}_1, \ldots, \mathrm{A}_k) = \mathrm{A}$.
  3. For a $\mathrm{PL^{nq}}$ formula A we show that $\mathcal{I}^\psi(\mathrm{A}) = \mathcal{I}_\psi(\mathrm{A})$ by induction on A.
     1. If $\mathrm{A} = p(\mathrm{A}_1, \ldots, \mathrm{A}_k)$, then $\mathcal{I}^\psi(\mathrm{A}) = \mathcal{I}^\psi(p)(\mathcal{I}(\mathrm{A}_1), \ldots, \mathcal{I}(\mathrm{A}_n)) = \mathsf{T}$, iff $\langle \mathrm{A}_1, \ldots, \mathrm{A}_k \rangle \in \mathcal{I}^\psi(p)$, iff $\psi(\theta_\psi^{-1}\mathrm{A}) = \mathsf{T}$, so $\mathcal{I}^\psi(\mathrm{A}) = \mathcal{I}_\psi(\mathrm{A})$ as desired.
     2. If $\mathrm{A} = \neg \mathrm{B}$, then $\mathcal{I}^\psi(\mathrm{A}) = \mathsf{T}$, iff $\mathcal{I}^\psi(\mathrm{B}) = \mathsf{F}$, iff $\mathcal{I}^\psi(\mathrm{B}) = \mathcal{I}_\psi(\mathrm{B})$, iff $\mathcal{I}^\psi(\mathrm{A}) = \mathcal{I}_\psi(\mathrm{A})$.
     3. If $\mathrm{A} = \mathrm{B} \wedge \mathrm{C}$ then we argue similarly
  4. Hence $\mathcal{I}^\psi(\mathrm{A}) = \mathcal{I}_\psi(\mathrm{A})$ for all $\mathrm{PL^{nq}}$ formulae and we have concluded the proof.

SOME RIGHTS RESERVED

5.1.3 Natural Language Semantics via Translation

SOME RIGHTS RESERVED

Translation rules for non-basic expressions (NP and S)

- **Definition 1.22.** We have the following translation rules for non-leaf node of the abstract syntax tree

  $T1$: $[X_{\mathrm{NP}}, Y_{V^i}]_S \rightsquigarrow Y'(X')$
  $T2$: $[X_{\mathrm{NP}}, Y_{V^t}, Z_{\mathrm{NP}}]_S \rightsquigarrow Y'(X', Z')$
  $T3$: $[X_{N_{\mathrm{pr}}}]_{\mathrm{NP}} \rightsquigarrow X'$
  $T4$: $[\mathrm{the}, X_N]_{\mathrm{NP}} \rightsquigarrow theX'$
  $T5$: $[\text{It is not the case that} X_S]_S \rightsquigarrow (\neg X')$
  $T6$: $[X_S, Y_{\mathrm{conj}}, Z_S]_S \rightsquigarrow Y'(X', Z')$
  $T7$: $[X_{\mathrm{NP}}, \mathrm{is}, Y_{\mathrm{NP}}]_S \rightsquigarrow X' = Y'$
  $T8$: $[X_{\mathrm{NP}}, \mathrm{is}, Y_{\mathrm{Adj}}]_S \rightsquigarrow Y'(X')$

  Read e.g. $[Y, Z]_X$ as a node with label X in the syntax tree with children X and Y. Read X' as the translation of X via these rules.
- Note that we have exactly one translation per syntax rule.

SOME RIGHTS RESERVED

Translation rule for basic lexical items

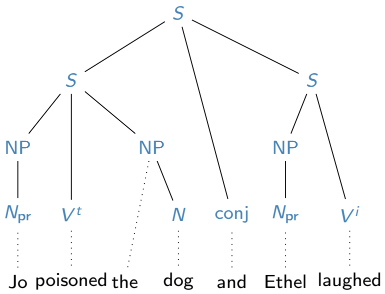
- **Definition 1.23.** The target logic for $\mathcal{F}_1$ is $\mathrm{PL}^{\mathrm{nq}}$, the fragment of PL^1 without quantifiers.
- **Lexical Translation Rules for $\mathcal{F}_1$ Categories:**
  - If w is a proper name, then $w' \in \Sigma_0^f$. (individual constant)
  - If w is an intransitive verb, then $w' \in \Sigma_1^p$. (one-place predicate)
  - If w is a transitive verb, $w' \in \Sigma_2^p$. (two-place predicate)
  - If w is a noun phrase, then $w' \in \Sigma_0^f$. (individual constant)
- **Semantics by Translation:** We translate sentences by translating their syntax trees via tree node translation rules.
- For any non-logical word w, we have the "pseudo-rule" $t1 : w \rightsquigarrow w'$.
- Note: This rule does not apply to the syncategorematic items is and the.
- Translations for logical connectives

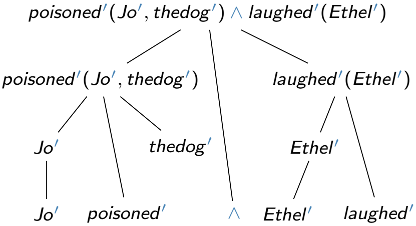
$$t2 : \text{and} \rightsquigarrow \wedge,\ t3 : \text{or} \rightsquigarrow \vee,\ t4 : \text{it is not the case that} \rightsquigarrow \neg$$

SOME RIGHTS RESERVED

Translation Example

- **Observation 1.24.** *Jo poisoned the dog and Ethel laughed is a sentence of Fragment 1*
- We can construct a syntax tree for it!





SOME RIGHTS RESERVED

5.2 Testing Truth Conditions via Inference

SOME RIGHTS RESERVED

Testing Truth Conditions in PL^{nq}

- **Idea 1:** To test our language model ($\mathcal{F}_1$)
  - Select a sentence S and a situation W that makes S true. (according to humans)
  - Translate S in to a formula S' in PL^{nq}.
  - Express W as a set Φ of formulae in PL^{nq} ($\Phi \mathrel{\widehat{=}}$ truth conditions)
  - Our language model is supported if $\Phi \models S'$, falsified if $\Phi \not\models S'$.
- **Example 2.1 (John chased the gangster in the red sports car).**
  - We claimed that we have three readings 3.9
    $R_1{:=}c(j,g) \wedge in(j,s)$, $R_2{:=}c(j,g) \wedge in(g,s)$, and $R_3{:=}c(j,g) \wedge in(j,s) \wedge in(g,s)$
  - So there must be three distinct situations W that make S true
    1. *John is in the red sports car, but the gangster isn't*
       $W_1{:=}c(j,g) \wedge in(j,s) \wedge \neg in(g,s)$, so $W_1 \models R_1$, but $W_1 \not\models R_2$ and $W_1 \not\models R_3$
    2. *The gangster is in the red sports car, but John isn't*
       $W_2{:=}c(j,g) \wedge in(j,s) \wedge \neg in(g,s)$, so $W_2 \models R_2$, but $W_2 \not\models R_1$ and $W_2 \not\models R_3$
    3. *Both are in the red sports car*
       $\widehat{=}$ they run around on the back seat of a very big sports car
       $W_3{:=}c(j,g) \wedge in(j,s) \wedge in(g,s)$, so $W_3 \models R_3$, but $W_3 \not\models R_1$ and $W_3 \not\models R_1$
- **Idea 2:** Use a calculus to model $\models$, e.g. $\mathcal{ND}$_0

SOME RIGHTS RESERVED

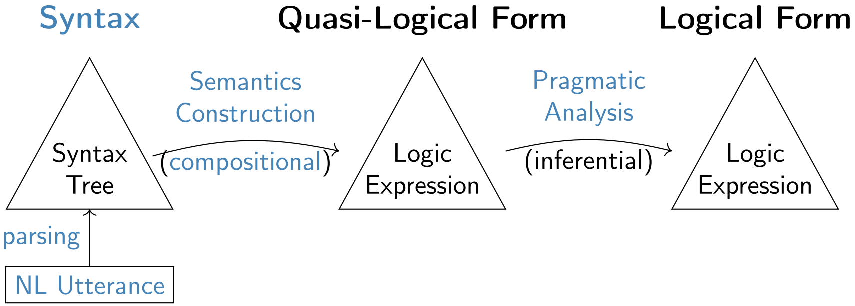
Fragment 1

- Fragment $\mathcal{F}_1$ of English (defined by grammar + lexicon)
- Logic PL^{nq} (serves as a mathematical model for $\mathcal{F}_1$)
  - Formal Language (individuals, predicates, $\neg, \wedge, \vee, \Rightarrow$)
  - Semantics $\mathcal{I}_\varphi$ defined recursively on formula structure ($\leadsto$ validity, entailment)
  - Tableau calculus for validity and entailment (Calculemus!)
- Analysis function $\mathcal{F}_1 \leadsto PL^{nq}$ (Translation)
- Test the model by checking predictions (calculate truth conditions)
- **Coverage:** Extremely Boring! (accounts for 0 examples from the intro) but the conceptual setup is fascinating

SOME RIGHTS RESERVED

Summary: The Interpretation Process

- **Interpretation Process:**



SOME RIGHTS RESERVED

Chapter 6
Fragment 1: The Grammatical Logical Framework

SOME RIGHTS RESERVED

6.1 Implementing Fragment 1 in GF

SOME RIGHTS RESERVED

Implementing Fragment 1 in GF

- The grammar of Fragment 1 only differs trivially from Hello World grammar `two.gf` from slide 65.
  - Verbs: $V^t \mathrel{\widehat{=}}$ `V2`, $V^i \mathrel{\widehat{=}}$ `cat V; fun sp : NP -> V -> S;`
  - Negation:
    `fun not : S -> S; lin not a = mkS ("it is not the case that"++ a.s);`
  - the: `fun the : N -> NP; lin the n = mkNP ("the"++ n.s);`
  - conjunction:
    `fun and : S -> S -> S; lin and a b = mkS (a.s ++ "and"++ b.s);`

SOME RIGHTS RESERVED

6.2 Implementing Fragment1 in GF and MMT

SOME RIGHTS RESERVED

Discourse Domain Theories for $\mathcal{F}_1$ (Lexicon)

- A "lexicon theory" (only selected constants here)

```
theory plnqFrag1 : ?plnq =
    ethel : ι | # ethel' ❙
    prudence : ι | # prudence' ❙
    dog : ι | # dog' ❙
    poison : ι → ι → o | # poison' 1 2 ❙
    laugh : ι → o | # laugh' 1 ❙
❚
```

  declares one logical constant for each from abstract GF grammar.

- Enough to interpret *Prudence poisoned the dog and Ethel laughed* from above.

```
    ex : | o = poison' prudence' dog' ∧ laugh' ethel' ❙
```


SOME RIGHTS RESERVED

Representing Multiple Readings

- We can even represent the three readings of *John chased the gangster in the red sports car* from 3.9.

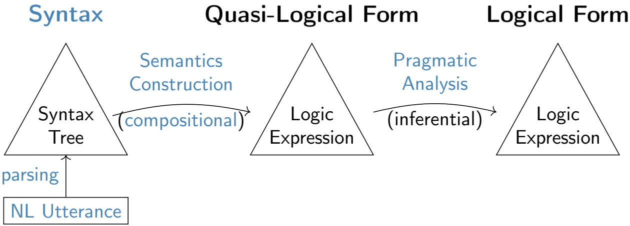
```
theory sportscar : ?plnq =
    john : ι ❚ gangster : ι ❚ sportscar : ι ❚ red : ι → o ❚
    chased : ι → ι → o ❚ in : ι → ι → o ❚
    jcgirs1 : o ❘= chased john gangster ∧ in sportscar gangster ∧
red sportscar ❚
    jcgirs2 : o ❘= chased john gangster ∧ in sportscar john ∧ red s
    jcgirs3 : o ❘= chased john gangster ∧ in sportscar john ∧
                                in sportscar gangster ∧ red sportscar ❚
❚
```

- **Problem:** Can we systematically generate terms like `jcgirs1`, `jcgirs2`, and `jcgirs3`?
- **Idea:** Use the ASTs from GF in Mmt.

SOME RIGHTS RESERVED

Embedding GF into Mmt

- **Observation:** The GF system provides Java bindings and Mmt is programed in Scala, which compiles into the Java virtual machine.
- **Idea:** Use GF as a sophisticated NL-parser/generator for Mmt
  - ⇝ Mmt with a natural language front-end.
  - ⇝ GF with a multi-logic back-end
- **Definition 2.1.** The MMT integration mapping interprets GF abstract syntax trees as Mmt terms.
- **Observation:** This fits very well with our interpretation process in LBS



- **Implementation:** transform GF system (Java) data structures to Mmt (Scala) ones in Mmt.

SOME RIGHTS RESERVED

GF Abstract syntax trees as Mmt Terms

- **Idea:** Make the MMT integration mapping (essentially) the identity.
- **Prerequisite:** Mmt theory isomorphic to GF grammar (declarations aligned)
- **Recall:** ASTs in GF are essentially terms.
- **Indeed:** GF abstract grammars are essentially Mmt theories.

SOME RIGHTS RESERVED

GF Abstract syntax trees as Mmt Terms

- **Idea:** Make the MMT integration mapping (essentially) the identity.
- **Prerequisite:** Mmt theory isomorphic to GF grammar (declarations aligned)
- **Recall:** ASTs in GF are essentially terms.
- **Indeed:** GF abstract grammars are essentially Mmt theories.
- **Example 2.3.** Syntactic categories of $\mathcal{F}_1$ (Syntactic categories $\widehat{=}$ types)

```
theory Frag1CatMMT : ur:?LF =
    S : type ❙
    Conj : type ❙
    NP : type ❙
    Npr : type ❙
    N : type ❙
    Vi : type ❙
    Vt : type ❙
❚
```

GF Abstract syntax trees as Mmt Terms

- **Idea:** Make the MMT integration mapping (essentially) the identity.
- **Prerequisite:** Mmt theory isomorphic to GF grammar (declarations aligned)
- **Recall:** ASTs in GF are essentially terms.
- **Indeed:** GF abstract grammars are essentially Mmt theories.
- **Example 2.4.** The $\mathcal{F}_1$ lexicon (words $\widehat{=}$ constants)

```
theory Frag1LexMMT : ur:?LF =
    include ? Frag1CatMMT
    ethel : Npr
    prudence : Npr
    dog : N
    poison : Vt
    laugh : Vi
    and : Conj
```


SOME RIGHTS RESERVED

GF Abstract syntax trees as Mmt Terms

- **Idea:** Make the MMT integration mapping (essentially) the identity.
- **Prerequisite:** Mmt theory isomorphic to GF grammar (declarations aligned)
- **Recall:** ASTs in GF are essentially terms.
- **Indeed:** GF abstract grammars are essentially Mmt theories.
- **Example 2.5.** The structural rules of $\mathcal{F}_1$ (functions $\widehat{=}$ functions)

```
theory Frag1RulesMMT : ur:?LF =
    include ? Frag1CatMMT
    s1 : NP → Vi → S ❙
    s2 : NP → Vt → NP → S ❙
    n1 : Npr → NP ❙
    n2 : N → NP ❙
    s3 : S → S ❙
    s4 : S → Conj → S → S ❙
    s5 : NP → NP → S ❙
    s6 : NP → Adj → S ❙
❚
```


SOME RIGHTS RESERVED

GF Abstract syntax trees as Mmt Terms

- **Idea:** Make the MMT integration mapping (essentially) the identity.
- **Prerequisite:** Mmt theory isomorphic to GF grammar (declarations aligned)
- **Recall:** ASTs in GF are essentially terms.
- **Indeed:** GF abstract grammars are essentially Mmt theories.
- **Example 2.6.** putting it all together

```
theory Frag1LexMMT : ur:?LF =
    include ? Frag1LexMMT
    include ? Frag1RulesMMT
```


SOME RIGHTS RESERVED

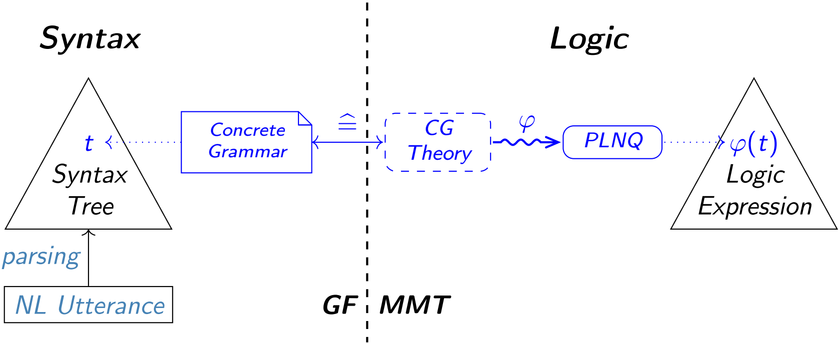
GF Abstract syntax trees as Mmt Terms

- **Idea:** Make the MMT integration mapping (essentially) the identity.
- **Prerequisite:** Mmt theory isomorphic to GF grammar (declarations aligned)
- **Recall:** ASTs in GF are essentially terms.
- **Indeed:** GF abstract grammars are essentially Mmt theories.
- **Observation:** GF grammars and Mmt theories best when organized modularly.
- **Example 2.7.**

SOME RIGHTS RESERVED

Semantics Construction as an MMT View

- **Observation 2.8.** *We can express semantics construction as an* Mmt *view*



- **Example 2.9.**

Semantics Construction as an MMT View

- **Observation 2.10.** *We can express semantics construction as an* Mmt *view*
- **Example 2.11.** Syntactic categories $\rightsquigarrow$ $\mathrm{PL^{nq}}$ types

```
view Frag1CatSem : ?Frag1CatMMT -> ?plnqFrag1 =
    S = o
    NP = ι
    Vi = ι → o
    Vt = ι → ι → o
    Npr = ι
    N = ι
    Conj = o → o → o
```


SOME RIGHTS RESERVED

Semantics Construction as an MMT View

- **Observation 2.12.** *We can express semantics construction as an* Mmt *view*
- **Example 2.13.** Lexicon $\leadsto$ mapping into $\mathrm{PL}^{\mathrm{nq}}$ terms

```
view Frag1LexSem : ?Frag1CatMMT -> ?plnqFrag1 =
    include ?Frag1CatSem
    ethel = ethel’ ❙
    prudence = prudence’ ❙
    dog = dog’ ❙
    poison = poison ❙
    laugh = laugh ❙
    and = and ❙
❚
```


SOME RIGHTS RESERVED

Semantics Construction as an MMT View

- **Observation 2.14.** *We can express semantics construction as an* Mmt *view*
- **Example 2.15.** Structural rules $\rightsquigarrow$ defining functions via λ-terms

```
view Frag1RulesSem : ?Frag1CatMMT -> ?plnqFrag1 =
    include ?Frag1CatSem
    s1 = [n, v] v n ❘
    s2 = [n1,v,n2] v n1 n2 ❘
    n1 = [n] n ❘
    n2 = [n] n ❘
    s3 = [s] ¬s ❘
    s4 = [a,c,b] c a b ❘
    s5 = [n1,n2] n1 ≐n2 ❘
    s6 = [n,a] a s ❘
❙
```


SOME RIGHTS RESERVED

Semantics Construction as an MMT View

- **Observation 2.16.** *We can express semantics construction as an* Mmt *view*
- **Example 2.17.** putting it all together

```
view Frag1Sem : ?Frag1CatMMT -> ?plnqFrag1 =
    include ?Frag1LexSem
    include ?Frag1RulesSem
```


SOME RIGHTS RESERVED

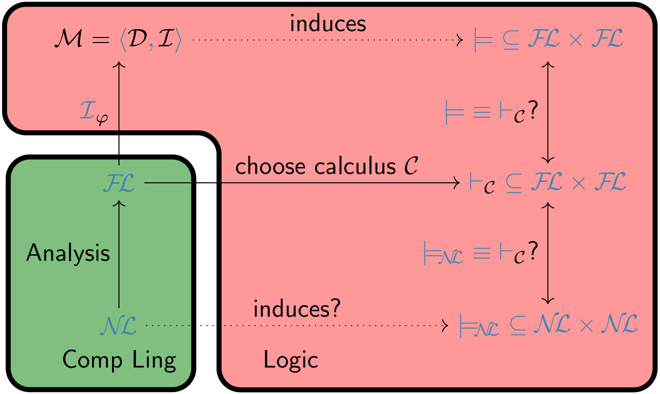
Montague-Style Processing of $\mathcal{F}_1$ in GLF

- **Example 2.18.** *Prudence poisoned the dog and Ethel laughed*
  - Parsing with GF
    - `parse -lang=Eng "Prudence poisons the dog and Ethel laughs"`
    - `s4 (s2 (n1 prudence) poison (n2 dog)) and (s1 (n1 ethel) laugh)`
  - Semantics construction via GLF: GF parsing + Mmt view
    - `parse -lang=Eng "Ethel poisons the dog and Prudence laughs"` construct|
    - `poison' prudence'` $\wedge$ `dog' laugh' ethel'`

SOME RIGHTS RESERVED

Montague-Style Analysis of $\mathcal{F}_1$ in GF and MMT

- **Recap:** We have realized the green part of



  - The GF grammar for $\mathcal{F}_1$ defines the fragment $\mathcal{NL}$.
  - The Mmt implementation of $\mathrm{PL}^{\mathrm{nq}}$ is $\mathcal{FL}$.
  - The Mmt view implements the compositional translation function for $\mathcal{F}_1$

SOME RIGHTS RESERVED

6.3 Implementing Natural Deduction in MMT

SOME RIGHTS RESERVED

Implementing Calculi in Mmt (Judgments as Types)

- **Idea:** Represent proofs and derivations as expressions in theory of "proofs" .
- **Concretely:** For any proposition A, introduce $\vdash$ A for *the type of proofs of A*.
  - Any term of type $\vdash$ A $\widehat{=}$ a *proof of A*
  - *A is provable* $\widehat{=}$ $\vdash$ A is nonempty
  - inference rules are proof constructors (functions)
  - a declaration `c : ⊢A` makes $\neg$`A` non-empty $\leadsto$ `c : ⊢A` $\widehat{=}$ an axiom
  - a definition `c : ⊢A |= P` does as well but also exhibits a "proof" `P` $\leadsto$ `c : ⊢A |= P` $\widehat{=}$ a theorem
- **in MMT:** we introduce a (proof) type constructor ded a type $\vdash$ A.

```
theory pl0NDminimal : ur:?LF =
    include ?proplogMinimal ❙
    ded : o → type | # ⊢1 prec 10 | role Judgment ❙
```

the `role` Judgment specifies ?????

SOME RIGHTS RESERVED

Implementing Calculi in Mmt ($\mathcal{ND}\_0$ Rules)

- **Recap:** We only need the $\mathcal{ND}\_0$ rules for negation and conjunction:

$$\frac{A \quad B}{A \wedge B}\mathcal{ND}\_0\wedge I \qquad \frac{A \wedge B}{A}\mathcal{ND}\_0\wedge E_l \qquad \frac{A \wedge B}{B}\mathcal{ND}\_0\wedge E_r \qquad \frac{\begin{array}{cc}[A]^1 & [A]^1\\ \vdots & \vdots\\ C & \neg C\end{array}}{\neg A}\mathcal{ND}\_0\neg I^1$$

- **The ND Rules:**

```
    notE  : {A} ⊢¬¬A  →⊢A | # ¬E 2 |
    notI  : {A,Q} (⊢A  →⊢Q) →(⊢A  →⊢¬Q) →⊢¬A | # ¬I 3 4 |
    andI  : {A,B} ⊢A  →⊢B  →⊢A∧ B | # ∧ I 3 4 |
    andEl  : {A,B} ⊢A∧ B  →⊢A | # ∧ El 3 |
    andEr  : {A,B} ⊢A∧ B  →⊢B | # ∧ Er 3 |
```

Inference rules as and hypothetical derivations as proof-to-proof functions.

- **Derived ND Rules:** All other inference rules of $\mathcal{ND}\_0$ can be written down similarly. What is more, as they are derivable from those above, they can become Mmt definitions.

SOME RIGHTS RESERVED

Implementing Calculi in Mmt (a proof)

- **Example 3.1.** We can now write down the proof for the commutativity of V!

$$\dfrac{\dfrac{\dfrac{[A \wedge B]^1}{B}\,\mathcal{ND}\_0 \wedge E_r \quad \dfrac{[A \wedge B]^1}{A}\,\mathcal{ND}\_0 \wedge E_l}{B) \wedge A}\,\mathcal{ND}\_0 \wedge I}{A \wedge B \Rightarrow B) \wedge A}\,\mathcal{ND}\_0 \Rightarrow I^1$$

from **??** as the Mmt declaration

```
andcomm {A,B} ⊢A∧ B ⇒ B∧ A | = ⇒ I([x] ∧ I (∧ Er x) (∧ El x))
```


SOME RIGHTS RESERVED

Chapter 7
Adding Context: Pronouns and World Knowledge

SOME RIGHTS RESERVED

7.1 Fragment 2: Pronouns and Anaphora

SOME RIGHTS RESERVED

Fragment 2 ($\mathcal{F}_2 \mathrel{\widehat{=}} \mathcal{F}_1$ + Pronouns)

- **Want to cover:** *Peter loves Fido. He bites him.* (almost intro)
  - **We need**: Translation and interpretation for *he*, *she*, *him*,. . . .
  - **Also**: A way to integrate world knowledge to filter out one interpretation (i.e. *Humans don't bite dogs.*)
- **Idea:** Integrate variables into $\mathrm{PL}^{\mathrm{nq}}$ (work backwards from that)
- **Logical System:** $\mathrm{PL}_{\mathrm{NQ}}^{\mathcal{V}} = \mathrm{PL}^{\mathrm{nq}} +$ variables (Translate pronouns to variables)

SOME RIGHTS RESERVED

New Grammar in $\mathcal{F}_2$ (Pronouns)

- **Definition 1.1.** We have the following structural grammar rules in $\mathcal{F}_2$

$$
\begin{aligned}
S1\colon & S \to \mathrm{NP}, V^i, \\
S2\colon & S \to \mathrm{NP}, V^t, \mathrm{NP}, \\
N1\colon & \mathrm{NP} \to N_{\mathrm{pr}}, \\
N2\colon & \mathrm{NP} \to \mathrm{Pron}, \\
N3\colon & \mathrm{NP} \to \mathrm{the}, N, \\
S3\colon & S \to \text{it is not the case that}, S, \\
S4\colon & S \to S, \mathrm{conj}, S, \\
S5\colon & S \to \mathrm{NP}, \mathrm{is}, \mathrm{NP}, \\
S6\colon & S \to \mathrm{NP}, \mathrm{is}, \mathrm{Adj}
\end{aligned}
$$

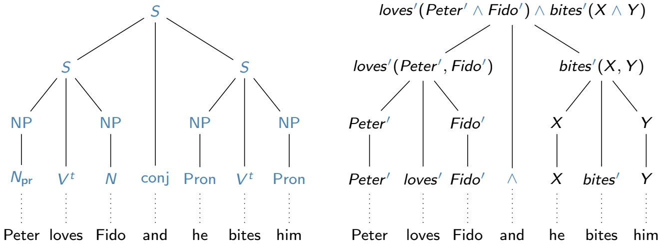
and one additional lexical rule:

$$L7\colon \mathrm{Pron} \to \mathit{he} \mid \mathit{she} \mid \mathit{it} \mid \mathit{we} \mid \mathit{they}$$

SOME RIGHTS RESERVED

Translation for $\mathcal{F}_2$ (first attempt)

- **Idea:** Pronouns are translated into new variables (so far)
- The syntax/semantic trees for *Peter loves Fido and he bites him.* are straightforward. (almost intro)



SOME RIGHTS RESERVED

Predicate Logic with Variables (but no Quantifiers)

- **Definition 1.2 (Logical System $\mathrm{PL}^{\mathcal{V}}_{\mathrm{NQ}}$).** $\mathrm{PL}^{\mathcal{V}}_{\mathrm{NQ}} := \mathrm{PL}^{\mathrm{nq}}$ + variables
- **Definition 1.3 ($\mathrm{PL}^{\mathcal{V}}_{\mathrm{NQ}}$ Syntax).**
  Category $\mathcal{V} = \{X, Y, Z, X^1, X^2, \ldots\}$ of variables (allow variables wherever individual constants were allowed)
- **Definition 1.4 ($\mathrm{PL}^{\mathcal{V}}_{\mathrm{NQ}}$ Semantics).** Model $\mathcal{M} = \langle \mathcal{D}, \mathcal{I} \rangle$ (need to evaluate variables)
  - variable assignment: $\varphi\colon \mathcal{V}_\iota \rightarrow U$
  - value function: $\mathcal{I}_\varphi(X) = \varphi(X)$ (defined like $\mathcal{I}$ elsewhere)
  - call a $\mathrm{PL}^{\mathcal{V}}_{\mathrm{NQ}}$ formula A valid in $\mathcal{M}$ under φ, iff $\mathcal{I}_\varphi(\mathrm{A}) = \mathsf{T}$,
  - call it satisfiable in $\mathcal{M}$, iff there is a variable assignment φ, such that $\mathcal{I}_\varphi(\mathrm{A}) = \mathsf{T}$

SOME RIGHTS RESERVED

Implementing Fragment 2 in GF

- The grammar of Fragment 2 only differs from that of Fragment 1 by
  - Pronouns: Pron $\widehat{=}$ `cat Pron; fun usePron : Pron -> NP; he,she,it : Pron;`,
  - Case: for distinguishing *he*/*him* in English.

```
param Case = nom | acc;
oper
  NounPhraseType : Type = { s : Case => Str };
  PronounType : Type = { s : Case => Str };
lincat
  NP = NounPhraseType;
  Pron = PronounType;
```

  - English Paradigms to deal with case

```
mkNP = overload {
  mkNP : Str -> NP =
      \name -> lin NP { s = table { nom => name; acc => name } };
  mkNP : (Case => Str) -> NP = \caseTable -> lin NP { s = caseTable };};
  mkPron : (she : Str) -> (her : Str) -> Pron =
      \she,her -> lin Pron {s = table {nom => she; acc => her}};
he = mkPron "he" "him" ; she = mkPron "she" "her";it = mkPron "it" "it";
```


SOME RIGHTS RESERVED

7.2 A Tableau Calculus for PLNQ with Free Variables

SOME RIGHTS RESERVED

7.2.1 Calculi for Automated Theorem Proving: Analytical Tableaux

SOME RIGHTS RESERVED

7.2.1.1 Analytical Tableaux

SOME RIGHTS RESERVED

Recap: Atoms and Literals

- **Definition 2.1.** A formula is called atomic (or an atom) if it does not contain logical constants, else it is called complex.
- **Definition 2.2.** We call a pair A^α of a formula and a truth value $\alpha \in \{T, F\}$ a labeled formula. For a set Φ of formulae we use $\Phi^\alpha := \{A^\alpha | A \in \Phi\}$.
- **Definition 2.3.** A labeled atom A^α is called a (positive if $\alpha = T$, else negative) literal.
- **Intuition:** To satisfy a formula, we make it "true". To satisfy a labeled formula A^α, it must have the truth value α.
- **Definition 2.4.** For a literal A^α, we call the literal A^β with $\alpha \neq \beta$ the opposite literal (or partner literal).

SOME RIGHTS RESERVED

Alternative Definition: Literals

- **Note:** Literals are often defined without recurring to labeled formulae:
- **Definition 2.5.** A literal is an atom A (positive literal) or negated atom $\neg$A (negative literal). A and $\neg$A are opposite literals.
- **Note:** This notion of literal is equivalent to the labeled formulae-notion of literal, but does not generalize as well to logics with more than two truth values.

SOME RIGHTS RESERVED

Test Calculi: Tableaux and Model Generation

- **Idea:** A tableau calculus is a test calculus that
  - analyzes a labeled formulae in a tree to determine satisfiability,
  - its branches correspond to valuations ($\leadsto$ models).
- **Example 2.6.** Tableau calculi try to construct models for labeled formulae:

Tableau refutation (Validity)	Model generation (Satisfiability)
$\models P \wedge Q \Rightarrow Q \wedge P$	$\models P \wedge (Q \vee \neg R) \wedge \neg Q$
$\begin{array}{c} (P \wedge Q \Rightarrow Q \wedge P)^{\mathsf{F}} \\ (P \wedge Q)^{\mathsf{T}} \\ (Q \wedge P)^{\mathsf{F}} \\ P^{\mathsf{T}} \\ Q^{\mathsf{T}} \\ \begin{array}{c\|c} P^{\mathsf{F}} & Q^{\mathsf{F}} \\ \bot & \bot \end{array} \end{array}$	$\begin{array}{c} (P \wedge (Q \vee \neg R) \wedge \neg Q)^{\mathsf{T}} \\ (P \wedge (Q \vee \neg R))^{\mathsf{T}} \\ \neg Q^{\mathsf{T}} \\ Q^{\mathsf{F}} \\ P^{\mathsf{T}} \\ (Q \vee \neg R)^{\mathsf{T}} \\ \begin{array}{c\|c} Q^{\mathsf{T}} & \neg R^{\mathsf{T}} \\ \bot & R^{\mathsf{F}} \end{array} \end{array}$
No Model	Herbrand Model $\{P^{\mathsf{T}}, Q^{\mathsf{F}}, R^{\mathsf{F}}\}$ $\varphi := \{P \mapsto \mathsf{T}, Q \mapsto \mathsf{F}, R \mapsto \mathsf{F}\}$

- **Idea:** Open branches in saturated tableaux yield models.
- **Algorithm:** Fully expand all possible tableaux, (no rule can be applied)
  - Satisfiable, iff there are open branches (correspond to models)

SOME RIGHTS RESERVED

Analytical Tableaux (Formal Treatment of $\mathcal{T}_0$)

- **Idea:** A test calculus where
  - A labeled formula is analyzed in a tree to determine satisfiability,
  - branches correspond to valuations (models)
- **Definition 2.7.** The propositional tableau calculus $\mathcal{T}_0$ has two inference rules per connective (one for each possible label)

$$\frac{(A \wedge B)^T}{\begin{array}{c}A^T \\ B^T\end{array}}\mathcal{T}_0\wedge \qquad \frac{(A \wedge B)^F}{A^F \mid B^F}\mathcal{T}_0\vee \qquad \frac{\neg A^T}{A^F}\mathcal{T}_0\neg^T \qquad \frac{\neg A^F}{A^T}\mathcal{T}_0\neg^F \qquad \frac{\begin{array}{c}A^\alpha \\ A^\beta\end{array} \quad \alpha \neq \beta}{\perp}\mathcal{T}_0\perp$$

Use rules exhaustively as long as they contribute new material ($\leadsto$ termination)

- **Definition 2.8.** We call any tree ($|$ introduces branches) produced by the $\mathcal{T}_0$ inference rules from a set Φ of labeled formulae a tableau for Φ.
- **Definition 2.9.** Call a tableau saturated, iff no rule adds new material and a branch closed, iff it ends in $\perp$, else open. A tableau is closed, iff all of its branches are.

SOME RIGHTS RESERVED

Analytical Tableaux ($\mathcal{T}_0$ continued)

- **Definition 2.10 ($\mathcal{T}_0$-Theorem/Derivability).** A is a $\mathcal{T}_0$-theorem ($\vdash_{\mathcal{T}_0} A$), iff there is a closed tableau with A^F at the root.
  $\Phi \subseteq wff_0(\mathcal{V}_0)$ derives A in $\mathcal{T}_0$ ($\Phi \vdash_{\mathcal{T}_0} A$), iff there is a closed tableau starting with A^F and Φ^T. The tableau with only a branch of A^F and Φ^T is called initial for $\Phi \vdash_{\mathcal{T}_0} A$.

SOME RIGHTS RESERVED

A Valid Real-World Example

- **Example 2.11.** *If Mary loves Bill and John loves Mary, then John loves Mary*

$$
\begin{array}{c}
(\mathrm{loves}(\mathrm{mary},\mathrm{bill}) \wedge \mathrm{loves}(\mathrm{john},\mathrm{mary}) \Rightarrow \mathrm{loves}(\mathrm{john},\mathrm{mary}))^F \\
\neg(\neg\neg(\mathrm{loves}(\mathrm{mary},\mathrm{bill}) \wedge \mathrm{loves}(\mathrm{john},\mathrm{mary})) \wedge \neg\mathrm{loves}(\mathrm{john},\mathrm{mary}))^F \\
(\neg\neg(\mathrm{loves}(\mathrm{mary},\mathrm{bill}) \wedge \mathrm{loves}(\mathrm{john},\mathrm{mary})) \wedge \neg\mathrm{loves}(\mathrm{john},\mathrm{mary}))^T \\
\neg\neg(\mathrm{loves}(\mathrm{mary},\mathrm{bill}) \wedge \mathrm{loves}(\mathrm{john},\mathrm{mary}))^T \\
\neg(\mathrm{loves}(\mathrm{mary},\mathrm{bill}) \wedge \mathrm{loves}(\mathrm{john},\mathrm{mary}))^F \\
(\mathrm{loves}(\mathrm{mary},\mathrm{bill}) \wedge \mathrm{loves}(\mathrm{john},\mathrm{mary}))^T \\
\neg\mathrm{loves}(\mathrm{john},\mathrm{mary})^T \\
\mathrm{loves}(\mathrm{mary},\mathrm{bill})^T \\
\mathrm{loves}(\mathrm{john},\mathrm{mary})^T \\
\mathrm{loves}(\mathrm{john},\mathrm{mary})^F \\
\bot
\end{array}
$$

This is a closed tableau, so the $\mathrm{loves}(\mathrm{mary},\mathrm{bill}) \wedge \mathrm{loves}(\mathrm{john},\mathrm{mary}) \Rightarrow \mathrm{loves}(\mathrm{john},\mathrm{mary})$ is a $\mathcal{T}_0$-theorem.

As we will see, $\mathcal{T}_0$ is sound and complete, so

$$\mathrm{loves}(\mathrm{mary},\mathrm{bill}) \wedge \mathrm{loves}(\mathrm{john},\mathrm{mary}) \Rightarrow \mathrm{loves}(\mathrm{john},\mathrm{mary})$$

is valid.

SOME RIGHTS RESERVED

Deriving Entailment in $\mathcal{T}_0$

- **Example 2.12.** *Mary loves Bill* and *John loves Mary* together entail that *John loves Mary*

$$\begin{array}{c} \mathrm{loves}(\mathrm{mary}, \mathrm{bill})^{\mathsf{T}} \\ \mathrm{loves}(\mathrm{john}, \mathrm{mary})^{\mathsf{T}} \\ \mathrm{loves}(\mathrm{john}, \mathrm{mary})^{\mathsf{F}} \\ \perp \end{array}$$

  This is a closed tableau, so
  $\{\mathrm{loves}(\mathrm{mary}, \mathrm{bill}), \mathrm{loves}(\mathrm{john}, \mathrm{mary})\} \vdash_{\mathcal{T}_0} \mathrm{loves}(\mathrm{john}, \mathrm{mary})$.
  Again, as $\mathcal{T}_0$ is sound and complete we have

$$\{\mathrm{loves}(\mathrm{mary}, \mathrm{bill}), \mathrm{loves}(\mathrm{john}, \mathrm{mary})\} \models \mathrm{loves}(\mathrm{john}, \mathrm{mary})$$

SOME RIGHTS RESERVED

A Falsifiable Real-World Example

- **Example 2.13.** * *If Mary loves Bill or John loves Mary, then John loves Mary*
  Try proving the implication (this fails)

$$
\begin{array}{c}
((\mathrm{loves}(\mathrm{mary},\mathrm{bill}) \lor \mathrm{loves}(\mathrm{john},\mathrm{mary})) \Rightarrow \mathrm{loves}(\mathrm{john},\mathrm{mary}))^{\mathsf{F}} \\
\neg(\neg\neg(\mathrm{loves}(\mathrm{mary},\mathrm{bill}) \lor \mathrm{loves}(\mathrm{john},\mathrm{mary})) \land \neg\mathrm{loves}(\mathrm{john},\mathrm{mary}))^{\mathsf{F}} \\
(\neg\neg(\mathrm{loves}(\mathrm{mary},\mathrm{bill}) \lor \mathrm{loves}(\mathrm{john},\mathrm{mary})) \land \neg\mathrm{loves}(\mathrm{john},\mathrm{mary}))^{\mathsf{T}} \\
\neg\mathrm{loves}(\mathrm{john},\mathrm{mary})^{\mathsf{T}} \\
\mathrm{loves}(\mathrm{john},\mathrm{mary})^{\mathsf{F}} \\
\neg\neg(\mathrm{loves}(\mathrm{mary},\mathrm{bill}) \lor \mathrm{loves}(\mathrm{john},\mathrm{mary}))^{\mathsf{T}} \\
\neg(\mathrm{loves}(\mathrm{mary},\mathrm{bill}) \lor \mathrm{loves}(\mathrm{john},\mathrm{mary}))^{\mathsf{F}} \\
(\mathrm{loves}(\mathrm{mary},\mathrm{bill}) \lor \mathrm{loves}(\mathrm{john},\mathrm{mary}))^{\mathsf{T}} \\
\begin{array}{c|c}
\mathrm{loves}(\mathrm{mary},\mathrm{bill})^{\mathsf{T}} & \mathrm{loves}(\mathrm{john},\mathrm{mary})^{\mathsf{T}} \\
 & \bot
\end{array}
\end{array}
$$

Indeed we can make $\mathcal{I}_\varphi(\mathrm{loves}(\mathrm{mary},\mathrm{bill})) = \mathsf{T}$ but $\mathcal{I}_\varphi(\mathrm{loves}(\mathrm{john},\mathrm{mary})) = \mathsf{F}$.

SOME RIGHTS RESERVED

Testing for Entailment in $\mathcal{T}_0$

- **Example 2.14.** Does *Mary loves Bill or John loves Mary* entail that *John loves Mary*?

$$
\begin{array}{c}
(\mathrm{loves}(\mathrm{mary},\mathrm{bill}) \lor \mathrm{loves}(\mathrm{john},\mathrm{mary}))^{\mathsf{T}} \\
\mathrm{loves}(\mathrm{john},\mathrm{mary})^{\mathsf{F}} \\
\begin{array}{c|c}
\mathrm{loves}(\mathrm{mary},\mathrm{bill})^{\mathsf{T}} & \mathrm{loves}(\mathrm{john},\mathrm{mary})^{\mathsf{T}} \\
 & \bot
\end{array}
\end{array}
$$

This saturated tableau has an open branch that shows that the interpretation with $\mathcal{I}_\varphi(\mathrm{loves}(\mathrm{mary},\mathrm{bill})) = \mathsf{T}$ but $\mathcal{I}_\varphi(\mathrm{loves}(\mathrm{john},\mathrm{mary})) = \mathsf{F}$ falsifies the derivability/entailment conjecture.

7.2.1.2 Practical Enhancements for Tableaux

SOME RIGHTS RESERVED

Derived Rules of Inference

- **Definition 2.15.** An inference rule $\frac{A_1 \quad \ldots \quad A_n}{C}$ is called derivable (or a derived rule) in a calculus $\mathcal{C}$, if there is a $\mathcal{C}$ derivation $A_1, \ldots, A_n \vdash_{\mathcal{C}} C$.
- **Definition 2.16.** We have the following derivable inference rules in $\mathcal{T}_0$:

$$\frac{(A \Rightarrow B)^T}{A^F \mid B^T} \qquad \frac{(A \Rightarrow B)^F}{\begin{array}{c} A^T \\ B^F \end{array}} \qquad \frac{\begin{array}{c} A^T \\ (A \Rightarrow B)^T \end{array}}{B^T}$$

$$\begin{array}{c} A^T \\ (A \Rightarrow B)^T \\ (\neg A \vee B)^T \\ \neg(\neg\neg A \wedge \neg B)^T \\ (\neg\neg A \wedge \neg B)^F \\ \begin{array}{c|c} \neg\neg A^F & \neg B^F \\ \neg A^T & B^T \\ A^F & \\ \bot & \end{array} \end{array}$$

$$\frac{(A \vee B)^T}{A^T \mid B^T} \qquad \frac{(A \vee B)^F}{\begin{array}{c} A^F \\ B^F \end{array}} \qquad \frac{A \Leftrightarrow B^T}{\begin{array}{c|c} A^T & A^F \\ B^T & B^F \end{array}} \qquad \frac{A \Leftrightarrow B^F}{\begin{array}{c|c} A^T & A^F \\ B^F & B^T \end{array}}$$

SOME RIGHTS RESERVED

Tableaux with derived Rules (example)

Example 2.17.

$$
\begin{array}{c}
(\mathsf{loves}(\mathsf{mary},\mathsf{bill}) \wedge \mathsf{loves}(\mathsf{john},\mathsf{mary}) \Rightarrow \mathsf{loves}(\mathsf{john},\mathsf{mary}))^{\mathsf{F}} \\
(\mathsf{loves}(\mathsf{mary},\mathsf{bill}) \wedge \mathsf{loves}(\mathsf{john},\mathsf{mary}))^{\mathsf{T}} \\
\mathsf{loves}(\mathsf{john},\mathsf{mary})^{\mathsf{F}} \\
\mathsf{loves}(\mathsf{mary},\mathsf{bill})^{\mathsf{T}} \\
\mathsf{loves}(\mathsf{john},\mathsf{mary})^{\mathsf{T}} \\
\bot
\end{array}
$$

SOME RIGHTS RESERVED

7.2.2 A Tableau Calculus for PLNQ with Free Variables

SOME RIGHTS RESERVED

A Tableau Calculus for $\mathrm{PL}^{\mathcal{V}}_{\mathrm{NQ}}$

- **Definition 2.18 (Tableau Calculus for $\mathbf{PL}^{\mathcal{V}}_{\mathbf{NQ}}$).** $\mathcal{T}^p_{\mathcal{V}} = \mathcal{T}_0$ + new tableau rules for formulae with variables

$$\frac{\begin{array}{c}\vdots \\ \mathrm{A}^{\alpha} \quad c\in\mathcal{H} \\ \vdots\end{array}}{([c/X](\mathrm{A}))^{\alpha}}\mathcal{T}^p_{\mathcal{V}}\,WK \qquad \frac{\begin{array}{cc}\vdots & \mathcal{H} = \{a_1, \ldots, a_n\} \\ \boxed{(\mathrm{A}^{\alpha})} & \mathrm{free}(\mathrm{A}) = \{X_1, \ldots, X_m\}\end{array}}{(\sigma_1(\mathrm{A}))^{\alpha} \,\Big|\, \ldots \,\Big|\, (\sigma_{(n^m)}(\mathrm{A}))^{\alpha}}\mathcal{T}^p_{\mathcal{V}}\,Ana$$

$\mathcal{H}$ is the set of ind. constants in the branch above (Herbrand Base) and the σ_i are substitutions that instantiate the X_j with any combinations of the a_k (there are n^m of them).

- the first rule is used for world knowledge (up in the branch)
- the second rule is used for input logical forms $\boxed{\cdots}$
  this rule has to be applied eagerly (while they are still at the leaf)

SOME RIGHTS RESERVED

Some Examples in $\mathcal{F}_2$

- **Example 2.19 (Peter snores).** (Only sleeping people snore)

$$\begin{array}{c} (\mathrm{snores}(X) \Rightarrow \mathrm{sleeps}(X))^{\mathsf{T}} \\ \boxed{(\mathrm{snores}(\mathrm{peter})^{\mathsf{T}})} \\ (\mathrm{snores}(\mathrm{peter}) \Rightarrow \mathrm{sleeps}(\mathrm{peter}))^{\mathsf{T}} \\ \mathrm{sleeps}(\mathrm{peter})^{\mathsf{T}} \end{array}$$

- **Example 2.20 (Peter sleeps. John walks. He snores).** (who snores?)

$$\begin{array}{c|c} \multicolumn{2}{c}{\boxed{(\mathrm{sleeps}(\mathrm{peter})^{\mathsf{T}})}} \\ \multicolumn{2}{c}{\boxed{(\mathrm{walks}(\mathrm{john})^{\mathsf{T}})}} \\ \multicolumn{2}{c}{\boxed{(\mathrm{snores}(X)^{\mathsf{T}})}} \\ \mathrm{snores}(\mathrm{peter})^{\mathsf{T}} & \mathrm{snores}(\mathrm{john})^{\mathsf{T}} \end{array}$$

SOME RIGHTS RESERVED

Does Tweety fly? The everlasting Question in AI

- **Example 2.21.**

Tweety is a bird

$$
\begin{array}{c|c}
\multicolumn{2}{c}{(\mathrm{bird}(X) \Rightarrow (\mathrm{flies}(X) \vee \mathrm{penguin}(X)))^{\mathsf{T}}} \\
\multicolumn{2}{c}{(\mathrm{penguin}(X) \Rightarrow \neg\mathrm{flies}(X))^{\mathsf{T}}} \\
\multicolumn{2}{c}{\boxed{(\mathrm{bird}(\mathrm{tweety})^{\mathsf{T}})}} \\
\multicolumn{2}{c}{(\mathrm{flies}(\mathrm{tweety}) \vee \mathrm{penguin}(\mathrm{tweety}))^{\mathsf{T}}} \\
\mathrm{flies}(\mathrm{tweety})^{\mathsf{T}} & \mathrm{penguin}(\mathrm{tweety})^{\mathsf{T}} \\
 & \neg\mathrm{flies}(\mathrm{tweety})^{\mathsf{T}} \\
 & \mathrm{flies}(\mathrm{tweety})^{\mathsf{F}}
\end{array}
$$

Tweety is an eagle

$$
\begin{array}{c|c}
\multicolumn{2}{c}{(\mathrm{bird}(X) \Rightarrow (\mathrm{flies}(X) \vee \mathrm{penguin}(X)))^{\mathsf{T}}} \\
\multicolumn{2}{c}{(\mathrm{eagle}(X) \Rightarrow \mathrm{bird}(X))^{\mathsf{T}}} \\
\multicolumn{2}{c}{(\mathrm{penguin}(X) \Rightarrow \neg\mathrm{eagle}(X))^{\mathsf{T}}} \\
\multicolumn{2}{c}{(\mathrm{penguin}(X) \Rightarrow \neg\mathrm{flies}(X))^{\mathsf{T}}} \\
\multicolumn{2}{c}{\boxed{(\mathrm{eagle}(\mathrm{tweety})^{\mathsf{T}})}} \\
\multicolumn{2}{c}{\mathrm{bird}(\mathrm{tweety})^{\mathsf{T}}} \\
\multicolumn{2}{c}{(\mathrm{flies}(\mathrm{tweety}) \vee \mathrm{penguin}(\mathrm{tweety}))^{\mathsf{T}}} \\
\mathrm{flies}(\mathrm{tweety})^{\mathsf{T}} & \mathrm{penguin}(\mathrm{tweety})^{\mathsf{T}} \\
 & (\neg\mathrm{eagle}(\mathrm{tweety}))^{\mathsf{T}} \\
 & \mathrm{eagle}(\mathrm{tweety})^{\mathsf{F}} \\
 & \bot
\end{array}
$$

- For the second we need to add more world knowledge.

SOME RIGHTS RESERVED

7.2.3 Case Study: Peter loves Fido, even though he sometimes bites him

SOME RIGHTS RESERVED

Finally: *Peter loves Fido. He bites him.*

- Let's try it naively (worry about the problems later.)

$$\boxed{(l(p,f)^{\mathsf{T}})}$$
$$\boxed{(b(X,Y)^{\mathsf{T}})}$$
$$b(p,p)^{\mathsf{T}} \mid b(p,f)^{\mathsf{T}} \mid b(f,p)^{\mathsf{T}} \mid b(f,f)^{\mathsf{T}}$$

- **Problem:** We get four readings instead of one!
- **Idea:** We have not specified enough world knowledge

SOME RIGHTS RESERVED

Peter and Fido with World Knowledge

- Nobody bites himself, humans do not bite dogs.

$$
\begin{array}{c}
d(f)^{T} \\
m(p)^{T} \\
b(X,X)^{F} \\
(d(X) \wedge m(Y) \Rightarrow \neg b(Y,X))^{T} \\
\boxed{(l(p,f)^{T})} \\
\boxed{(b(X,Y)^{T})} \\
\begin{array}{c|c|c|c}
b(p,p)^{T} & b(p,f)^{T} & b(f,p)^{T} & b(f,f)^{T} \\
b(p,p)^{F} & (d(f) \wedge m(p) \Rightarrow \neg b(p,f))^{T} & & b(f,f)^{F} \\
\bot & b(p,f)^{F} & & \bot \\
 & \bot & &
\end{array}
\end{array}
$$

- **Observation:** Pronoun resolution introduces ambiguities.
- **Pragmatics:** Use world knowledge to filter out impossible readings.

SOME RIGHTS RESERVED

7.2.4 The Computational Role of Ambiguities

SOME RIGHTS RESERVED

The computational Role of Ambiguities

- **Observation:** (in the traditional waterfall model)
  Every processing stage introduces ambiguities that need to be resolved.
  - Syntax: e.g. *Peter chased the man in the red sports car* (attachment)
  - Semantics: e.g. *Peter went to the bank* (lexical)
  - Pragmatics: e.g. *Two men carried two bags* (collective vs. distributive)
- **Question:** Where does pronoun-ambiguity belong? (much less clear)
- **Answer:** we have freedom to choose
  1. resolve the pronouns in the syntax (generic waterfall model)
     - $\leadsto$ multiple syntactic representations (pragmatics as filter)
  2. resolve the pronouns in the pragmatics (our model here)
     - $\leadsto$ need underspecified syntactic representations (e.g. variables)
     - $\leadsto$ pragmatics needs ambiguity treatment (e.g. tableaux)

SOME RIGHTS RESERVED

Translation for $\mathcal{F}_2$ Reconsidered

- **Idea:** Pronouns are translated into new variables (so far)
- **Problem:** *Peter loves Mary. She loves him.*

$$\boxed{(\mathrm{loves}(\mathrm{peter},\mathrm{mary})^{\mathsf{T}})}$$

$$\boxed{(\mathrm{loves}(X,Y)^{\mathsf{T}})}$$

$$\mathrm{loves}(\mathrm{peter},\mathrm{peter})^{\mathsf{T}} \mid \mathrm{loves}(\mathrm{peter},\mathrm{mary})^{\mathsf{T}} \mid \mathrm{loves}(\mathrm{mary},\mathrm{peter})^{\mathsf{T}} \mid \mathrm{loves}(\mathrm{mary},\mathrm{mary})$$

- **Idea:** attach world knowledge to pronouns (just as with Peter and Fido)
  - use the world knowledge to distinguish (linguistic) gender by predicates masc and fem
- **Idea:** attach world knowledge to pronouns (just as with Peter and Fido)
- **Problem:** properties of
  - proper names are given in the model,
  - pronouns must be given by the syntax/semantics interface
- **In particular:** How to generate $\mathrm{loves}(X,Y) \wedge \mathrm{masc}(X) \wedge \mathrm{fem}(Y)$ compositionally?

SOME RIGHTS RESERVED

Sorts refine World Categories

- **Definition 2.22 (Sorted Logics).** (in our case $PL^1_{\mathcal{S}}$) assume a set of sorts $\mathcal{S}:=\{\mathbb{A},\mathbb{B},\mathbb{C},\ldots\}$, annotate every syntactic and semantic structure with them. Make all constructions and operations well worted:
  - Syntax: variables and constants are sorted $X_{\mathbb{A}}, Y_{\mathbb{B}}, Z^1_{\mathbb{C}_1} \ldots, a_{\mathbb{A}}, b_{\mathbb{A}}, \ldots$
  - Semantics: subdivide the Universe $\mathcal{D}_\iota$ into subsets $\mathcal{D}_{\mathbb{A}} \subseteq \mathcal{D}_\iota$
    Interpretation $\mathcal{I}$ and variable assignment φ have to be well-sorted.
    $\mathcal{I}(a_{\mathbb{A}}), \varphi(X_{\mathbb{A}}) \in \mathcal{D}_{\mathbb{A}}$.
  - calculus: substitutions must be well sorted $[a_{\mathbb{A}}/X_{\mathbb{A}}]$ OK, $[a_{\mathbb{A}}/X_{\mathbb{B}}]$ not.
- **Observation:** Sorts do not add expressivity in principle (just practically) For every sort $\mathbb{A}$, we introduce a first-order predicate $\mathcal{R}_{\mathbb{A}}$ and
  - Translate $R(X_{\mathbb{A}}) \wedge \neg P(Z_{\mathbb{C}})$ to $\mathcal{R}_{\mathbb{A}}(X) \wedge \mathcal{R}_{\mathbb{C}}(Z) \Rightarrow R(X) \wedge \neg P(Z)$ in world knowledge.
  - Translate $R(X_{\mathbb{A}}) \wedge \neg P(Z_{\mathbb{C}})$ to $\mathcal{R}_{\mathbb{A}}(X) \wedge \mathcal{R}_{\mathbb{C}}(Z) \wedge R(X,Y) \wedge \neg P(Z)$ in input.
  - Meaning is preserved, but translation is non-compositional!

SOME RIGHTS RESERVED

7.3 Tableaux and Model Generation

SOME RIGHTS RESERVED

7.3.1 Tableau Branches and Herbrand Models

SOME RIGHTS RESERVED

Model Generation and Interpretation

- **Example 3.1 (from above).** In 2.14 we claimed that

$$\mathcal{H}:=\{\mathsf{loves}(\mathsf{john},\mathsf{mary})^{\mathsf{F}},\mathsf{loves}(\mathsf{mary},\mathsf{bill})^{\mathsf{T}}\}$$

  constitutes a model

$$\begin{array}{c|c} \multicolumn{2}{c}{(\mathsf{loves}(\mathsf{mary},\mathsf{bill})\vee\mathsf{loves}(\mathsf{john},\mathsf{mary}))^{\mathsf{T}}} \\ \multicolumn{2}{c}{\mathsf{loves}(\mathsf{john},\mathsf{mary})^{\mathsf{F}}} \\ \mathsf{loves}(\mathsf{mary},\mathsf{bill})^{\mathsf{T}} & \mathsf{loves}(\mathsf{john},\mathsf{mary})^{\mathsf{T}} \\ & \bot \end{array}$$

- **Recap:** A model $\mathcal{M}$ is a pair $\langle U,\mathcal{I}\rangle$, where $\mathcal{D}$ is a set of individuals, and $\mathcal{I}$ is an interpretation function.
- **Problem:** Find U and $\mathcal{I}$

Model Generation and Models

- **Idea:** Choose the universe U as the set Σ_0^f of constants, choose $\mathcal{I}(=)\mathrm{Id}_{\Sigma_0^f}$, interpret $p\in\Sigma_k^p$ via $\mathcal{I}(p):=\{\langle a_1, \ldots, a_k\rangle | p(a_1, \ldots, a_k)\in\mathcal{H}\}$.
- **Definition 3.2.** We call a model a Herbrand model, iff $U = \Sigma_0^f$ and $\mathcal{I} = \mathrm{Id}_{\Sigma_0^f}$.
- **Lemma 3.3.**
  Let $\mathcal{H}$ be a set of atomic propositions, then setting

  $$\mathcal{I}(p):=\{\langle a_1, \ldots, a_k\rangle | p(a_1, \ldots, a_k)\in\mathcal{H}\}$$

  yields a Herbrand Model that satisfies $\mathcal{H}$. (*proof trivial*)
- **Corollary 3.4.** *Let $\mathcal{H}$ be a consistent (i.e. ∇_c holds) set of atomic propositions, then there is a Herbrand Model that satisfies $\mathcal{H}$.* (*take $\mathcal{H}^{\mathrm{T}}$*)

SOME RIGHTS RESERVED

7.3.2 Using Model Generation for Interpretation

SOME RIGHTS RESERVED

Using Model Generation for Interpretation

- **Definition 3.5.** Mental model theory [JL83; JLB91] posits that humans form mental models of the world, i.e. (neural) representations of possible states of the world that are consistent with the perceptions up to date and use them to reason about the world.
- **So** communication by natural language is a process of transporting parts of the mental model of the speaker into the mental model of the hearer.
- **Therefore** the NL interpretation process on the part of the hearer is a process of integrating the meaning of the utterances of the speaker into his mental model.
- **Idea:** We can model discourse understanding as a process of generating Herbrand models for the logical form of an utterance in a discourse by a tableau based model generation procedure.
- **Advantage:** Capturing ambiguity by generating multiple models for input logical forms.

SOME RIGHTS RESERVED

Tableau Machine

- **Definition 3.6.** The tableau machine is an inferential cognitive model for incremental natural language understanding that implements mental model theory via tableau based model generation over a sequence of input sentences.
  It iterates the following process for every input sentence staring with the empty tableau:
  1. add the logical form of the input sentence S_i to the selected branch,
  2. perform tableau inferences below S_i until saturated or some resource criterion is met
  3. if there are open branches choose a "preferred branch", otherwise backtrack to previous tableau for S_j with $j < i$ and open branches, then re-process $S_{j+1}, \ldots, S_i$ if possible, else fail.

  The output is application dependent; some choices are
  - the Herbrand model for the preferred branch $\leadsto$ preferred interpretation;
  - the literals augmented with all non expanded formulae (from the discourse); (resource-bound was reached)
  - machine answers user queries (preferred model $\models$ query?)
- model generation mode (guided by resources and strategies)
- theorem proving mode ($\Box$ for side conditions; using tableau rules)

SOME RIGHTS RESERVED

The Tableau Machine in Action

- **Example 3.7.** The tableau machine in action on two sentences.

initialize tableau

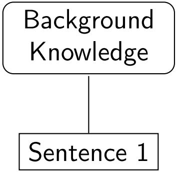
Background
Knowledge

SOME RIGHTS RESERVED

The Tableau Machine in Action

- **Example 3.8.** The tableau machine in action on two sentences.

input sentence

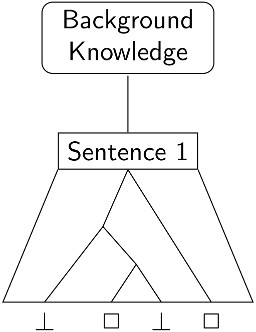


SOME RIGHTS RESERVED

The Tableau Machine in Action

- **Example 3.9.** The tableau machine in action on two sentences.

saturate tableau

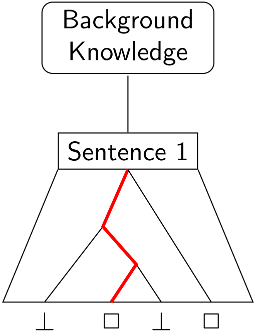


SOME RIGHTS RESERVED

The Tableau Machine in Action

- **Example 3.10.** The tableau machine in action on two sentences.

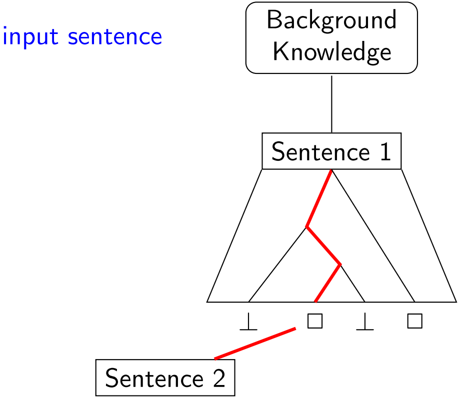
choose branch



SOME RIGHTS RESERVED

The Tableau Machine in Action

- **Example 3.11.** The tableau machine in action on two sentences.

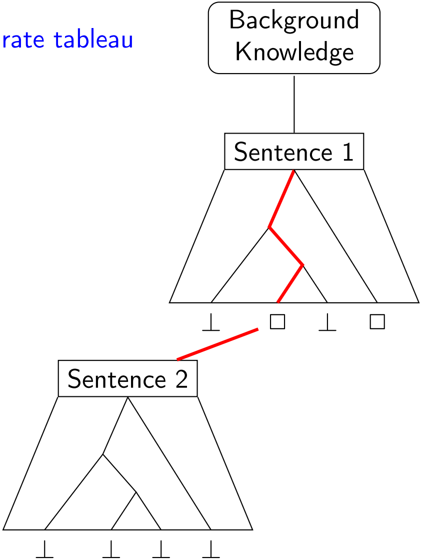


SOME RIGHTS RESERVED

The Tableau Machine in Action

▶ **Example 3.12.** The tableau machine in action on two sentences.

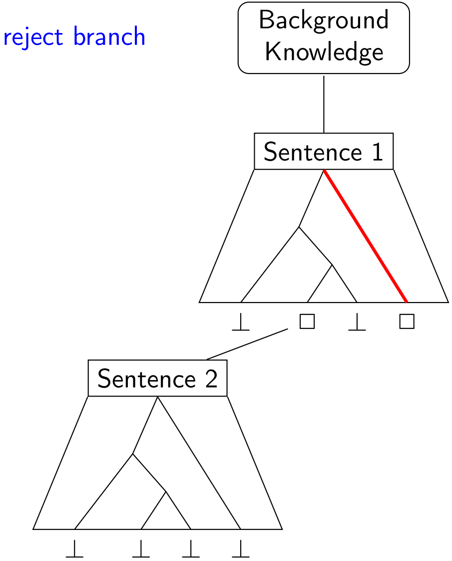
saturate tableau



SOME RIGHTS RESERVED

The Tableau Machine in Action

- **Example 3.13.** The tableau machine in action on two sentences.

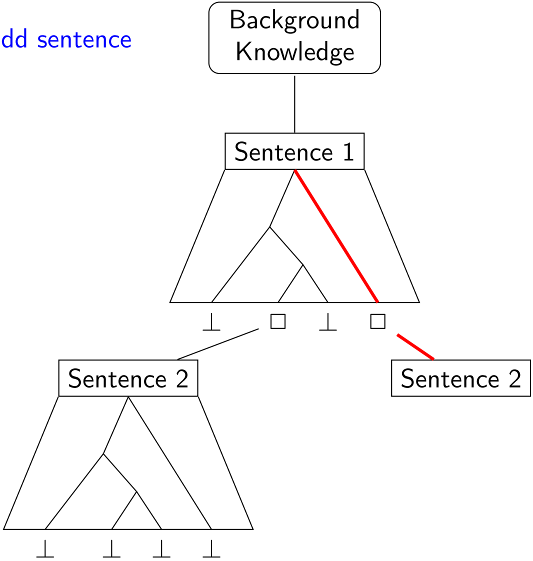


SOME RIGHTS RESERVED

The Tableau Machine in Action

▶ **Example 3.14.** The tableau machine in action on two sentences.

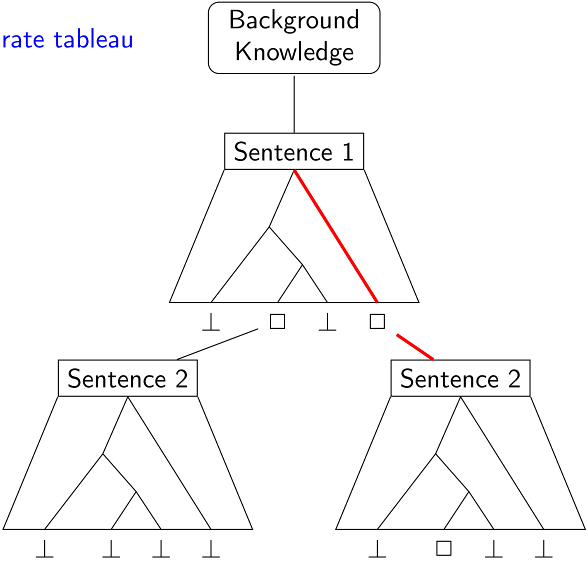
re-add sentence



The Tableau Machine in Action

- **Example 3.15.** The tableau machine in action on two sentences.

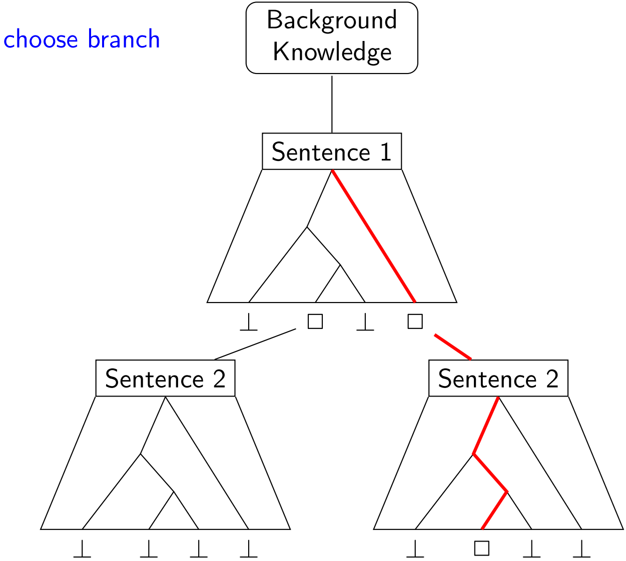
saturate tableau



SOME RIGHTS RESERVED

The Tableau Machine in Action

- **Example 3.16.** The tableau machine in action on two sentences.



Two (Syntactical) Readings

- **Example 3.17.** *Peter loves Mary and Mary sleeps or Peter snores* (syntactically ambiguous)

  Reading 1 $\mathrm{loves}(\mathrm{peter},\mathrm{mary}) \wedge (\mathrm{sleeps}(\mathrm{mary}) \vee \mathrm{snores}(\mathrm{peter}))$

  Reading 2 $\mathrm{loves}(\mathrm{peter},\mathrm{mary}) \wedge \mathrm{sleeps}(\mathrm{mary}) \vee \mathrm{snores}(\mathrm{peter})$
- Let us first consider the first reading in 3.17. Let us furthermore assume that we start out with the empty tableau, even though this is cognitively implausible, since it simplifies the presentation.

$$\begin{array}{c|c}
\multicolumn{2}{c}{\boxed{\mathrm{loves}(\mathrm{peter},\mathrm{mary}) \wedge (\mathrm{sleeps}(\mathrm{mary}) \vee \mathrm{snores}(\mathrm{peter}))}} \\
\multicolumn{2}{c}{\mathrm{loves}(\mathrm{peter},\mathrm{mary})^{\mathsf{T}}} \\
\multicolumn{2}{c}{(\mathrm{sleeps}(\mathrm{mary}) \vee \mathrm{snores}(\mathrm{peter}))^{\mathsf{T}}} \\
\mathrm{sleeps}(\mathrm{mary})^{\mathsf{T}} & \mathrm{snores}(\mathrm{peter})^{\mathsf{T}}
\end{array}$$

- **Observation:** We have two models, so we have a case of semantical ambiguity.

SOME RIGHTS RESERVED

The other (Syntactical) Reading

$$\boxed{\mathsf{loves}(\mathsf{peter},\mathsf{mary})\land\mathsf{sleeps}(\mathsf{mary})\lor\mathsf{snores}(\mathsf{peter})}$$

$$\begin{array}{c|c} (\mathsf{loves}(\mathsf{peter},\mathsf{mary})\land\mathsf{sleeps}(\mathsf{mary}))^{\mathsf{T}} & \mathsf{snores}(\mathsf{peter})^{\mathsf{T}} \\ \mathsf{loves}(\mathsf{peter},\mathsf{mary})^{\mathsf{T}} & \\ \mathsf{sleeps}(\mathsf{mary})^{\mathsf{T}} & \end{array}$$

SOME RIGHTS RESERVED

Continuing the Discourse

- **Example 3.18.** *Peter does not love Mary*
  then the second tableau would be extended to

$$
\begin{array}{c|c}
\multicolumn{2}{c}{\boxed{\mathrm{loves}(\mathrm{peter},\mathrm{mary}) \wedge \mathrm{sleeps}(\mathrm{mary}) \vee \mathrm{snores}(\mathrm{peter})}} \\
(\mathrm{loves}(\mathrm{peter},\mathrm{mary}) \wedge \mathrm{sleeps}(\mathrm{mary}))^{\mathsf{T}} & \mathrm{snores}(\mathrm{peter})^{\mathsf{T}} \\
\mathrm{loves}(\mathrm{peter},\mathrm{mary})^{\mathsf{T}} & \boxed{\neg\mathrm{loves}(\mathrm{peter},\mathrm{mary})} \\
\mathrm{sleeps}(\mathrm{mary})^{\mathsf{T}} & \\
\boxed{\neg\mathrm{loves}(\mathrm{peter},\mathrm{mary})} & \\
\mathrm{loves}(\mathrm{peter},\mathrm{mary})^{\mathsf{F}} & \\
\bot &
\end{array}
$$

  and the first tableau closes altogether.
- In effect the choice of models has been reduced to one, which constitutes the intuitively correct reading of the discourse

SOME RIGHTS RESERVED

Model Generation models Discourse Understanding

- Conforms with psycholinguistic findings:
  - Zwaan& Radvansky [ZR98]: listeners not only represent logical form, but also models containing referents.
  - deVega [de 95]: online, incremental process.
  - Singer [Sin94]: enriched by background knowledge.
  - Glenberg et al. [GML87]: major function is to provide basis for anaphor resolution.

SOME RIGHTS RESERVED

Towards a Performance Model for NLU

- **Problem:** The tableau machine is only a competence model.
- **Definition 3.19.** A competence model is a meaning theory that delineates a space of possible discourses. A performance model delineates the discourses actually used in communication. (after [Cho65a])
- **Idea:** We need to guide the tableau machine in which inferences and branch choices it performs.
- **Idea:** Each tableau rule comes with rule costs.
  - **Here**: each sentence in the discourse has a fixed inference budget.
    Expansion until budget used up.
  - **Ultimately** we want bounded optimization regime [Rus91]:
    Expansion as long as expected gain in model quality outweighs proof costs
- **Effect:** Expensive rules are rarely applied. (only if the promise great rewards)
- ⚠ Finding appropriate values for rule costs and model quality is an open problem.

SOME RIGHTS RESERVED

7.3.3 Adding Equality to PLNQ or Fragment 1

SOME RIGHTS RESERVED

PL_{NQ}^ $=$: Adding Equality to PL^{nq}

- **Syntax:** Just another binary predicate constant $=$
- **Semantics:** Fixed as $\mathcal{I}_\varphi(a=b)=\mathsf{T}$, iff $\mathcal{I}_\varphi(a)=\mathcal{I}_\varphi(b)$. (logical constant)
- **Definition 3.20 (Tableau Calculus $\mathcal{T}_{NQ}^{=}$).** Add two additional inference rules (a positive and a negative) to $\mathcal{T}_0$

$$\frac{a\in\mathcal{H}}{a=a^{\mathsf{T}}}\,\mathcal{T}_{NQ}^{=}\mathrm{sym} \qquad \frac{\begin{array}{c}a=b^{\mathsf{T}}\\ \mathrm{A}\,[a]_p{}^{\alpha}\end{array}}{[b/p]\mathrm{A}^{\alpha}}\,\mathcal{T}_{NQ}^{=}\mathrm{rep}$$

  where
  - $\mathcal{H} \mathrel{\widehat{=}}$ the Herbrand Base, i.e. the set of constants occurring on the branch
  - we write $\mathrm{C}\,[\mathrm{A}]_p$ to indicate that $\mathrm{C}|_p=\mathrm{A}$ (C has subterm A at position p).
  - $[\mathrm{A}/p]\mathrm{C}$ is obtained from C by replacing the subterm at position p with A.
- **Note:** We could have equivalently written $\mathcal{T}_{NQ}^{=}\mathrm{sym}$ as $\frac{a=a^{\mathsf{F}}}{\perp}$: With $\mathcal{T}_{NQ}^{=}\mathrm{sym}$ we can conjure a $a=a^{\mathsf{T}}$ from thin air which can then be used to close the $a=a^{\mathsf{F}}$.
- **So, ...** $\mathcal{T}_{NQ}^{=}\mathrm{sym}$ and $\mathcal{T}_{NQ}^{=}\mathrm{rep}$ follow the pattern of having a T and a F rule per logical constant.

SOME RIGHTS RESERVED

Reading Comprehension Example: Mini TOEFL test

- **Example 3.21 (Reading Comprehension).** If you hear/read *Mary is the teacher. Peter likes the teacher.*, do you know whether *Peter likes Mary*?
- **Idea:** Interpret via tableau machine (interpretation mode) and test entailment in theorem proving mode.
- **Interpretation:** Feed $\Phi_1 := \mathrm{mary} = \mathrm{the\_teacher}$ and $\Phi_2 := \mathrm{likes}(\mathrm{peter}, \mathrm{the\_teacher})$ to the tableau machine in turn.
  Model generation tableau (nothing to do on these inputs)

$$\boxed{\mathrm{mary} = \mathrm{the\_teacher}^{\mathsf{T}}}$$
$$\boxed{\mathrm{likes}(\mathrm{peter}, \mathrm{the\_teacher})^{\mathsf{T}}}$$

- **Entailment Test:** label $\varphi := \mathrm{likes}(\mathrm{peter}, \mathrm{mary})$ with F and saturate the tableau.

$$\boxed{\mathrm{mary} = \mathrm{the\_teacher}^{\mathsf{T}}}$$
$$\boxed{\mathrm{likes}(\mathrm{peter}, \mathrm{the\_teacher})^{\mathsf{T}}}$$
$$\mathrm{likes}(\mathrm{peter}, \mathrm{mary})^{\mathsf{F}}$$
$$\mathrm{likes}(\mathrm{peter}, \mathrm{the\_teacher})^{\mathsf{F}}$$
$$\bot$$

Indeed, it closes, so $\Phi_1, \Phi_2 \models \varphi$.

SOME RIGHTS RESERVED

Chapter 8
Pronouns and World Knowledge in First-Order Logic

SOME RIGHTS RESERVED

8.1 First-Order Logic

SOME RIGHTS RESERVED

First-Order Predicate Logic (PL^1)

- **Coverage:** We can talk about (*All humans are mortal*)
  - individual things and denote them by variables or constants
  - properties of individuals, (e.g. being human or mortal)
  - relations of individuals, (e.g. *sibling_of* relationship)
  - functions on individuals, (e.g. the *father_of* function)

  We can also state the existence of an individual with a certain property, or the universality of a property.
- But we cannot state assertions like
  - *There is a surjective function from the natural numbers into the reals*.
- First-Order Predicate Logic has many good properties (complete calculi, compactness, unitary, linear unification,. . .)
- But too weak for formalizing: (at least directly)
  - natural numbers, torsion groups, calculus, . . .
  - generalized quantifiers (*most, few,. . .*)

SOME RIGHTS RESERVED

8.1.1 First-Order Logic: Syntax and Semantics

SOME RIGHTS RESERVED

PL^1 Syntax (Signature and Variables)

- **Definition 1.1.** First-order logic (PL^1), is a formal system extensively used in mathematics, philosophy, linguistics, and computer science. It combines propositional logic with the ability to quantify over individuals.
- PL^1 talks about two kinds of objects: (so we have two kinds of symbols)
  - truth values by reusing PL^0
  - individuals, e.g. numbers, foxes, Pokémon,...
- **Definition 1.2.** A first-order signature consists of (all disjoint; $k \in \mathbb{N}$)
  - connectives: $\Sigma_0 = \{T, F, \neg, \vee, \wedge, \Rightarrow, \Leftrightarrow, \ldots\}$ (functions on truth values)
  - function constants: $\Sigma_k^f = \{f, g, h, \ldots\}$ (k-ary functions on individuals)
  - predicate constants: $\Sigma_k^p = \{p, q, r, \ldots\}$ (k-ary relations among individuals.)
  - (Skolem constants: $\Sigma_k^{sk} = \{f_k^1, f_k^2, \ldots\}$) (witness constructors; countably ∞)
  - We take Σ_1 to be all of these together: $\Sigma_1 := \Sigma^f \cup \Sigma^p \cup \Sigma^{sk}$ and define $\Sigma := \Sigma_1 \cup \Sigma_0$.
- **Definition 1.3.** We assume a set of individual variables: $\mathcal{V}_\iota := \{X, Y, Z, \ldots\}$. (countably ∞)

SOME RIGHTS RESERVED

PL^1 Syntax (Formulae)

- **Definition 1.4.** Terms: $A \in wff_\iota(\Sigma_1, \mathcal{V}_\iota)$ (denote individuals)
  - $\mathcal{V}_\iota \subseteq wff_\iota(\Sigma_1, \mathcal{V}_\iota)$,
  - if $f \in \Sigma_k^f$ and $A^i \in wff_\iota(\Sigma_1, \mathcal{V}_\iota)$ for $i \leq k$, then $f(A^1, \ldots, A^k) \in wff_\iota(\Sigma_1, \mathcal{V}_\iota)$.
- **Definition 1.5.** Propositions: $A \in wff_o(\Sigma_1, \mathcal{V}_\iota)$: (denote truth values)
  - if $p \in \Sigma_k^p$ and $A^i \in wff_\iota(\Sigma_1, \mathcal{V}_\iota)$ for $i \leq k$, then $p(A^1, \ldots, A^k) \in wff_o(\Sigma_1, \mathcal{V}_\iota)$,
  - if $A, B \in wff_o(\Sigma_1, \mathcal{V}_\iota)$ and $X \in \mathcal{V}_\iota$, then $T, A \wedge B, \neg A, \forall X.A \in wff_o(\Sigma_1, \mathcal{V}_\iota)$.
    $\forall$ is a binding operator called the universal quantifier.
- **Definition 1.6.** We define the connectives $F, \vee, \Rightarrow, \Leftrightarrow$ via the abbreviations $A \vee B := \neg(\neg A \wedge \neg B)$, $A \Rightarrow B := \neg A \vee B$, $A \Leftrightarrow B := (A \Rightarrow B) \wedge (B \Rightarrow A)$, and $F := \neg T$. We will use them like the primary connectives $\wedge$ and $\neg$
- **Definition 1.7.** We use $\exists X.A$ as an abbreviation for $\neg(\forall X.\neg A)$. $\exists$ is a binding operator called the existential quantifier.
- **Definition 1.8.** Call formulae without connectives or quantifiers atomic else complex.

SOME RIGHTS RESERVED

Alternative Notations for Quantifiers

Here	Elsewhere	
$\forall x.A$	$\bigwedge x.A$	$(x)A$
$\exists x.A$	$\bigvee x.A$	

SOME RIGHTS RESERVED

Free and Bound Variables

- **Definition 1.9.** We call an occurrence of a variable X bound in a formula A, iff it occurs in a sub-formula $\forall X.\mathrm{B}$ of A. We call a variable occurrence free otherwise.
  For a formula A, we will use BVar(A) (and free(A)) for the set of bound (free) variables of A, i.e. variables that have a free/bound occurrence in A.
- **Definition 1.10.** We define the set free(A) of frees variable of a formula A:

$$
\begin{aligned}
&\mathrm{free}(X){:=}\{X\}\\
&\mathrm{free}(f(\mathrm{A}_1,\ldots,\mathrm{A}_n)){:=}\bigcup_{1\leq i\leq n}\mathrm{free}(\mathrm{A}_i)\\
&\mathrm{free}(p(\mathrm{A}_1,\ldots,\mathrm{A}_n)){:=}\bigcup_{1\leq i\leq n}\mathrm{free}(\mathrm{A}_i)\\
&\mathrm{free}(\neg\mathrm{A}){:=}\mathrm{free}(\mathrm{A})\\
&\mathrm{free}(\mathrm{A}\wedge\mathrm{B}){:=}\mathrm{free}(\mathrm{A})\cup\mathrm{free}(\mathrm{B})\\
&\mathrm{free}(\forall X.\mathrm{A}){:=}\mathrm{free}(\mathrm{A})\backslash\{X\}
\end{aligned}
$$

- **Definition 1.11.** We call a formula A closed or ground, iff $\mathrm{free}(\mathrm{A})=\emptyset$. We call a closed proposition a sentence, and denote the set of all ground terms with $cwff_\iota(\Sigma_1)$ and the set of sentences with $cwff_o(\Sigma_1)$.

SOME RIGHTS RESERVED

Semantics of PL^1 (Models)

- **Definition 1.12.** We inherit the domain $\mathcal{D}_0 = \{\mathsf{T}, \mathsf{F}\}$ of truth values from PL^0 and assume an arbitrary domain $\mathcal{D}_\iota \neq \emptyset$ of individuals.(this choice is a parameter to the semantics)
- **Definition 1.13.** An interpretation $\mathcal{I}$ assigns values to constants, e.g.
  - $\mathcal{I}(\neg)\colon \mathcal{D}_0 \to \mathcal{D}_0$ with $\mathsf{T} \mapsto \mathsf{F}$, $\mathsf{F} \mapsto \mathsf{T}$, and $\mathcal{I}(\wedge) = \ldots$ (as in PL^0)
  - $\mathcal{I}\colon \Sigma_k^f \to {\mathcal{D}_\iota}^k \to \mathcal{D}_\iota$ (interpret function symbols as arbitrary functions)
  - $\mathcal{I}\colon \Sigma_k^p \to \mathcal{P}({\mathcal{D}_\iota}^k)$ (interpret predicates as arbitrary relations)
- **Definition 1.14.** A variable assignment $\varphi\colon \mathcal{V}_\iota \to \mathcal{D}_\iota$ maps variables into the domain.
- **Definition 1.15.** A model $\mathcal{M} = \langle \mathcal{D}_\iota, \mathcal{I} \rangle$ of PL^1 consists of a domain $\mathcal{D}_\iota$ and an interpretation $\mathcal{I}$.

SOME RIGHTS RESERVED

Semantics of PL^1 (Evaluation)

- **Definition 1.16.** Given a model $\langle\mathcal{D},\mathcal{I}\rangle$, the value function $\mathcal{I}_\varphi$ is recursively defined: (two parts: terms & propositions)
  - $\mathcal{I}_\varphi\colon wff_\iota(\Sigma_1,\mathcal{V}_\iota)\to\mathcal{D}_\iota$ assigns values to terms.
    - $\mathcal{I}_\varphi(X):=\varphi(X)$ and
    - $\mathcal{I}_\varphi(f(\mathrm{A}_1,\ldots,\mathrm{A}_k)):=\mathcal{I}(f)(\mathcal{I}_\varphi(\mathrm{A}_1),\ldots,\mathcal{I}_\varphi(\mathrm{A}_k))$
  - $\mathcal{I}_\varphi\colon wff_o(\Sigma_1,\mathcal{V}_\iota)\to\mathcal{D}_0$ assigns values to formulae:
    - $\mathcal{I}_\varphi(T)=\mathcal{I}(T)=\mathsf{T}$,
    - $\mathcal{I}_\varphi(\neg\mathrm{A})=\mathcal{I}(\neg)(\mathcal{I}_\varphi(\mathrm{A}))$
    - $\mathcal{I}_\varphi(\mathrm{A}\wedge\mathrm{B})=\mathcal{I}(\wedge)(\mathcal{I}_\varphi(\mathrm{A}),\mathcal{I}_\varphi(\mathrm{B}))$ (just as in PL^0)
    - $\mathcal{I}_\varphi(p(\mathrm{A}_1,\ldots,\mathrm{A}_k)):=\mathsf{T}$, iff $\langle\mathcal{I}_\varphi(\mathrm{A}_1),\ldots,\mathcal{I}_\varphi(\mathrm{A}_k)\rangle\in\mathcal{I}(p)$
    - $\mathcal{I}_\varphi(\forall X.\mathrm{A}):=\mathsf{T}$, iff $\mathcal{I}_{(\varphi,[\mathsf{a}/X])}(\mathrm{A})=\mathsf{T}$ for all $\mathsf{a}\in\mathcal{D}_\iota$.
- **Definition 1.17 (Assignment Extension).** Let φ be a variable assignment into D and $a\in D$, then $\varphi,[a/X]$ is called the extension of φ with $[a/X]$ and is defined as $\{(Y,a)\in\varphi|Y\neq X\}\cup\{(X,a)\}$: $\varphi,[a/X]$ coincides with φ off X, and gives the result a there.

SOME RIGHTS RESERVED

Semantics Computation: Example

- **Example 1.18.** We define an instance of first-order logic:
  - Signature: Let $\Sigma_0^f := \{j, m\}$, $\Sigma_1^f := \{f\}$, and $\Sigma_2^p := \{o\}$
  - Universe: $\mathcal{D}_\iota := \{J, M\}$
  - Interpretation: $\mathcal{I}(j) := J$, $\mathcal{I}(m) := M$, $\mathcal{I}(f)(J) := M$, $\mathcal{I}(f)(M) := M$, and $\mathcal{I}(o) := \{(M, J)\}$.

  Then $\forall X.o(f(X), X)$ is a sentence and with $\psi := \varphi, [a/X]$ for $a \in \mathcal{D}_\iota$ we have

$$
\begin{array}{rll}
\mathcal{I}_\varphi(\forall X.o(f(X), X)) = \mathsf{T} & \text{iff} & \mathcal{I}_\psi(o(f(X), X)) = \mathsf{T} \text{ for all } a \in \mathcal{D}_\iota \\
& \text{iff} & (\mathcal{I}_\psi(f(X)), \mathcal{I}_\psi(X)) \in \mathcal{I}(o) \text{ for all } a \in \{J, M\} \\
& \text{iff} & (\mathcal{I}(f)(\mathcal{I}_\psi(X)), \psi(X)) \in \{(M, J)\} \text{ for all } a \in \{J, M\} \\
& \text{iff} & (\mathcal{I}(f)(\psi(X)), a) = (M, J) \text{ for all } a \in \{J, M\} \\
& \text{iff} & \mathcal{I}(f)(a) = M \text{ and } a = J \text{ for all } a \in \{J, M\}
\end{array}
$$

  But $a \neq J$ for $a = M$, so $\mathcal{I}_\varphi(\forall X.o(f(X), X)) = \mathsf{F}$ in the model $\langle \mathcal{D}_\iota, \mathcal{I} \rangle$.

SOME RIGHTS RESERVED

8.1.2 First-Order Substitutions

SOME RIGHTS RESERVED

Substitutions on Terms

- **Intuition:** If B is a term and X is a variable, then we denote the result of systematically replacing all occurrences of X in a term A by B with $[\mathrm{B}/X](\mathrm{A})$.
- **Problem:** What about $[Z/Y],[Y/X](X)$, is that Y or Z?
- **Folklore:** $[Z/Y],[Y/X](X)=Y$, but $[Z/Y]([Y/X](X))=Z$ of course. (Parallel application)
- **Definition 1.19.** Let $wfe(\Sigma,\mathcal{V})$ be an expression language, then we call $\sigma\colon\mathcal{V}\to wfe(\Sigma,\mathcal{V})$ a substitution, iff the support $\mathrm{supp}(\sigma):=\{X|(X,\mathrm{A})\in\sigma, X\neq\mathrm{A}\}$ of σ is finite. We denote the empty substitution with ϵ.
- **Definition 1.20 (Substitution Application).** We define substitution application by
  - $\sigma(c)=c$ for $c\in\Sigma$
  - $\sigma(X)=\mathrm{A}$, iff $\mathrm{A}\in\mathcal{V}$ and $(X,\mathrm{A})\in\sigma$.
  - $\sigma(f(\mathrm{A}_1,\ldots,\mathrm{A}_n))=f(\sigma(\mathrm{A}_1),\ldots,\sigma(\mathrm{A}_n))$,
  - $\sigma(\beta X.\mathrm{A})=\beta X.\sigma_{-X}(\mathrm{A})$.
- **Example 1.21.** $[a/x],[f(b)/y],[a/z]$ instantiates $g(x,y,h(z))$ to $g(a,f(b),h(a))$.
- **Definition 1.22.** Let σ be a substitution then we call $\mathrm{intro}(\sigma):=\bigcup_{X\in\mathrm{supp}(\sigma)}\mathrm{free}(\sigma(X))$ the set of variables introduced by σ.

SOME RIGHTS RESERVED

Substitution Extension

- **Definition 1.23 (Substitution Extension).**
  Let σ be a substitution, then we denote the extension of σ with $[\mathrm{A}/X]$ by $\sigma,[\mathrm{A}/X]$ and define it as $\{(Y,\mathrm{B}) \in \sigma \mid Y \neq X\} \cup \{(X,\mathrm{A})\}$: $\sigma,[\mathrm{A}/X]$ coincides with σ off X, and gives the result A there.
- **Note:** If σ is a substitution, then $\sigma,[\mathrm{A}/X]$ is also a substitution.
- We also need the dual operation: removing a variable from the support:
- **Definition 1.24.** We can discharge a variable X from a substitution σ by setting $\sigma_{-X} := \sigma,[X/X]$.

SOME RIGHTS RESERVED

Substitutions on Propositions

- **Problem:** We want to extend substitutions to propositions, in particular to quantified formulae: What is $\sigma(\forall X.\mathrm{A})$?
- **Idea:** σ should not instantiate bound variables. ($[\mathrm{A}/X](\forall X.\mathrm{B}) = \forall \mathrm{A}.\mathrm{B}'$ ill-formed)
- **Definition 1.25.** $\sigma(\forall X.\mathrm{A}):=(\forall X.\sigma_{-X}(\mathrm{A}))$.
- **Problem:** This can lead to variable capture: $[f(X)/Y](\forall X.p(X,Y))$ would evaluate to $\forall X.p(X,f(X))$, where the second occurrence of X is bound after instantiation, whereas it was free before.
- **Definition 1.26.** Let $\mathrm{B}\in wf\!f_\iota(\Sigma_\iota,\mathcal{V}_\iota)$ and $\mathrm{A}\in wf\!f_o(\Sigma_\iota,\mathcal{V}_\iota)$, then we call B substitutable for X in A, iff A has no occurrence of X in a subterm $\forall Y.\mathrm{C}$ with $Y\in\mathrm{free}(\mathrm{B})$.
- **Solution:** Forbid substitution $[\mathrm{B}/X]\mathrm{A}$, when B is not substitutablex for X in A.
- **Better Solution:** Rename away the bound variable X in $\forall X.p(X,Y)$ before applying the substitution. (see alphabetic renaming later.)

SOME RIGHTS RESERVED

Substitution Value Lemma for Terms

- **Lemma 1.27.** *Let* A *and* B *be terms, then* $\mathcal{I}_\varphi([\mathrm{B}/X]\mathrm{A}) = \mathcal{I}_\psi(\mathrm{A})$, *where* $\psi = \varphi, [\mathcal{I}_\varphi(\mathrm{B})/X]$.
- *Proof:* by induction on the depth of A:
  1. depth=0 *Then* A *is a variable (say* Y*), or constant, so we have three cases*
     1.1. $\mathrm{A} = Y = X$
     1.1.1. then $\mathcal{I}_\varphi([\mathrm{B}/X](\mathrm{A})) = \mathcal{I}_\varphi([\mathrm{B}/X](X)) = \mathcal{I}_\varphi(\mathrm{B}) = \psi(X) = \mathcal{I}_\psi(X) = \mathcal{I}_\psi(\mathrm{A})$.
     1.2. $\mathrm{A} = Y \neq X$
     1.2.1. then $\mathcal{I}_\varphi([\mathrm{B}/X](\mathrm{A})) = \mathcal{I}_\varphi([\mathrm{B}/X](Y)) = \mathcal{I}_\varphi(Y) = \varphi(Y) = \psi(Y) = \mathcal{I}_\psi(Y) = \mathcal{I}_\psi(\mathrm{A})$.
     1.3. A is a constant
     1.3.1. Analogous to the preceding case ($Y \neq X$).
     1.4. This completes the base case (depth = 0).
  2. depth> 0
     2.1. then $\mathrm{A} = f(\mathrm{A}_1, \ldots, \mathrm{A}_n)$ and we have

$$\begin{aligned}\mathcal{I}_\varphi([\mathrm{B}/X](\mathrm{A})) &= \mathcal{I}(f)(\mathcal{I}_\varphi([\mathrm{B}/X](\mathrm{A}_1)), \ldots, \mathcal{I}_\varphi([\mathrm{B}/X](\mathrm{A}_n))) \\ &= \mathcal{I}(f)(\mathcal{I}_\psi(\mathrm{A}_1), \ldots, \mathcal{I}_\psi(\mathrm{A}_n)) \\ &= \mathcal{I}_\psi(\mathrm{A}).\end{aligned}$$

by induction hypothesis

SOME RIGHTS RESERVED

Substitution Value Lemma for Propositions

- **Lemma 1.28.** *Let* $\mathrm{B}\in wf\!f_\iota(\Sigma_\iota,\mathcal{V}_\iota)$ *be substitutable for* X *in* $\mathrm{A}\in wf\!f_o(\Sigma_\iota,\mathcal{V}_\iota)$, *then* $\mathcal{I}_\varphi([\mathrm{B}/X](\mathrm{A}))=\mathcal{I}_\psi(\mathrm{A})$, *where* $\psi=\varphi,[\mathcal{I}_\varphi(\mathrm{B})/X]$.
- *Proof:* by induction on the number n of connectives and quantifiers in A
  1. $n=0$
     1.1. then A is an atomic proposition, and we can argue like in the induction step of the substitution value lemma for terms.
  2. $n>0$ and $\mathrm{A}=\neg\mathrm{B}$ or $\mathrm{A}=\mathrm{C}\circ\mathrm{D}$
     2.1. Here we argue like in the induction step of the term lemma as well.
  3. $n>0$ and $\mathrm{A}=\forall X.\mathrm{C}$
     3.1. then $\mathcal{I}_\psi(\mathrm{A})=\mathcal{I}_\psi(\forall X.\mathrm{C})=\mathsf{T}$, iff $\mathcal{I}_{(\psi,[a/X])}(\mathrm{C})=\mathcal{I}_{(\varphi,[a/X])}(\mathrm{C})=\mathsf{T}$, for all $a\in\mathcal{D}_\iota$, which is the case, iff $\mathcal{I}_\varphi(\forall X.\mathrm{C})=\mathcal{I}_\varphi([\mathrm{B}/X](\mathrm{A}))=\mathsf{T}$.
  4. $n>0$ and $\mathrm{A}=\forall Y.\mathrm{C}$ where $X\neq Y$
     4.1. then $\mathcal{I}_\psi(\mathrm{A})=\mathcal{I}_\psi(\forall Y.\mathrm{C})=\mathsf{T}$, iff $\mathcal{I}_{(\psi,[a/Y])}(\mathrm{C})=\mathcal{I}_{(\varphi,[a/Y])}([\mathrm{B}/X](\mathrm{C}))=\mathsf{T}$, by induction hypothesis.
     4.2. So $\mathcal{I}_\psi(\mathrm{A})=\mathcal{I}_\varphi(\forall Y.[\mathrm{B}/X](\mathrm{C}))=\mathcal{I}_\varphi([\mathrm{B}/X](\forall Y.\mathrm{C}))=\mathcal{I}_\varphi([\mathrm{B}/X](\mathrm{A}))$

SOME RIGHTS RESERVED

8.1.3 Alpha-Renaming for First-Order Logic

SOME RIGHTS RESERVED

Alphabetic Renaming

- **Lemma 1.29.** *Bound variables can be renamed: If Y is substitutable for X in* A, *then* $\mathcal{I}_\varphi(\forall X.\mathrm{A}) = \mathcal{I}_\varphi(\forall Y.[Y/X](\mathrm{A}))$.
- *Proof:* by the definitions:
  1. $\mathcal{I}_\varphi(\forall X.\mathrm{A}) = \mathsf{T}$, iff
  2. $\mathcal{I}_{(\varphi,[a/X])}(\mathrm{A}) = \mathsf{T}$ for all $a \in \mathcal{D}_\iota$, iff
  3. $\mathcal{I}_{(\varphi,[a/Y])}([Y/X](\mathrm{A})) = \mathsf{T}$ for all $a \in \mathcal{D}_\iota$, iff (by substitution value lemma)
  4. $\mathcal{I}_\varphi(\forall Y.[Y/X](\mathrm{A})) = \mathsf{T}$.
- **Definition 1.30.** We call two formulae A and B alphabetic variants (or α-equal; write $\mathrm{A} =_\alpha \mathrm{B}$), iff $\mathrm{A} = \forall X.\mathrm{C}$ and $\mathrm{B} = \forall Y.[Y/X](\mathrm{C})$ for some variables X and Y.

SOME RIGHTS RESERVED

Avoiding Variable Capture by Built-in α-renaming

- **Idea:** Given alphabetic renaming, consider alphabetic variants as identical!
- **So:** Bound variable names in formulae are just a representational device. (we rename bound variables wherever necessary)
- **Formally:** Take $cwff_o(\Sigma_\iota)$ (new) to be the quotient space of $cwff_o(\Sigma_\iota)$ (old) modulo $=_\alpha$. (formulae as syntactic representatives of equivalence classes)
- **Definition 1.31 (Capture-Avoiding Substitution Application).** Let σ be a substitution, A a formula, and A′ an alphabetic variant of A, such that $\mathrm{intro}(\sigma) \cap \mathrm{BVar}(\mathrm{A}) = \emptyset$. Then $[\mathrm{A}]_{=_\alpha} = [\mathrm{A}']_{=_\alpha}$ and we can define $\sigma([\mathrm{A}]_{=_\alpha}) := [(\sigma(\mathrm{A}'))]_{=_\alpha}$.
- **Notation:** After we have understood the quotient construction, we will neglect making it explicit and write formulae and substitutions with the understanding that they act on quotients.
- **Alternative:** Replace variables with numbers in formulae (de Bruijn indices).

SOME RIGHTS RESERVED

Undecidability of First-Order Logic

- **Theorem 1.32.** *Validity in first-order logic is undecidable.*
- *Proof:* We prove this by contradiction
  1. Let us assume that there is a

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG SOME RIGHTS RESERVED

8.2 First-Order Inference with Tableaux

SOME RIGHTS RESERVED

First-Order Standard Tableaux ($\mathcal{T}_1$)

- **Definition 2.1.** The standard tableau calculus ($\mathcal{T}_1$) extends $\mathcal{T}_0$ (propositional tableau calculus) with the following quantifier rules:

$$\frac{(\forall X.\mathrm{A})^{\mathsf{T}} \quad \mathrm{C}\in cwff_{\iota}(\Sigma_{\iota})}{([\mathrm{C}/X](\mathrm{A}))^{\mathsf{T}}}\mathcal{T}_1\forall \qquad \frac{(\forall X.\mathrm{A})^{\mathsf{F}} \quad c\in\Sigma_0^{sk} \text{ new}}{([c/X](\mathrm{A}))^{\mathsf{F}}}\mathcal{T}_1\exists$$

- **Problem:** The rule $\mathcal{T}_1\forall$ displays a case of "don't know indeterminism": to find a refutation we have to guess a formula C from the (usually infinite) set $cwff_{\iota}(\Sigma_{\iota})$.
For proof search, this means that we have to systematically try all, so $\mathcal{T}_1\forall$ is infinitely branching in general.

SOME RIGHTS RESERVED

8.2.1 Free Variable Tableaux

SOME RIGHTS RESERVED

Free variable Tableaux ($\mathcal{T}_1^f$)

- **Definition 2.2.** The free variable tableau calculus ($\mathcal{T}_1^f$) extends $\mathcal{T}_0$ (propositional tableau calculus) with the quantifier rules:

$$\frac{(\forall X.A)^{\mathsf{T}} \quad Y \text{ new}}{([Y/X](A))^{\mathsf{T}}}\mathcal{T}_1^f\forall \qquad \frac{(\forall X.A)^{\mathsf{F}} \quad \mathrm{free}(\forall X.A) = \{X^1, \ldots, X^k\} \quad f \in \Sigma_k^{sk} \text{ new}}{([f(X^1, \ldots, X^k)/X](A))^{\mathsf{F}}}\mathcal{T}_1^f\exists$$

  and generalizes its cut rule $\mathcal{T}_0\bot$ to:

$$\frac{\begin{array}{l} A^\alpha \\ B^\beta \end{array} \quad \alpha \neq \beta \quad \sigma(A) = \sigma(B)}{\bot : \sigma}\mathcal{T}_1^f\bot$$

  $\mathcal{T}_1^f\bot$ instantiates the whole tableau by σ.

- **Advantage:** No guessing necessary in $\mathcal{T}_1^f\forall$-rule!
- **New Problem:** find suitable substitution (most general unifier) (later)

SOME RIGHTS RESERVED

Free variable Tableaux ($\mathcal{T}_1^f$): Derivable Rules

- **Definition 2.3.** Derivable quantifier rules in $\mathcal{T}_1^f$:

$$\frac{(\exists X.A)^{\mathsf{T}} \quad \mathrm{free}(\forall X.A) = \{X^1, \ldots, X^k\} \quad f \in \Sigma_k^{sk} \text{ new}}{([f(X^1, \ldots, X^k)/X](A))^{\mathsf{T}}}$$

$$\frac{(\exists X.A)^{\mathsf{F}} \quad Y \text{ new}}{([Y/X](A))^{\mathsf{F}}}$$

SOME RIGHTS RESERVED

Termination and Multiplicity in Tableaux

- **Recall:** In $\mathcal{T}_0$, all rules only needed to be applied once.
  $\leadsto$ $\mathcal{T}_0$ terminates and thus induces a decision procedure for PL^0.
- **Observation 2.4.** *All $\mathcal{T}_1^f$ rules except $\mathcal{T}_1^f\forall$ only need to be applied once.*

SOME RIGHTS RESERVED

Termination and Multiplicity in Tableaux

- **Recall:** In $\mathcal{T}_0$, all rules only needed to be applied once.
  $\leadsto$ $\mathcal{T}_0$ terminates and thus induces a decision procedure for PL^0.
- **Observation 2.9.** *All $\mathcal{T}_1^f$ rules except $\mathcal{T}_1^f\forall$ only need to be applied once.*
- **Example 2.10.** A tableau proof for $(p(a) \vee p(b)) \Rightarrow (\exists.p())$.

Start, close left branch	use $\mathcal{T}_1^f\forall$ again (right branch)
$((p(a) \vee p(b)) \Rightarrow (\exists.p()))^F$	$((p(a) \vee p(b)) \Rightarrow (\exists.p()))^F$
$(p(a) \vee p(b))^T$	$(p(a) \vee p(b))^T$
$(\exists x.p(x))^F$	$(\exists x.p(x))^F$
$(\forall x.\neg p(x))^T$	$(\forall x.\neg p(x))^T$
$\neg p(y)^T$	$\neg p(a)^T$
$p(y)^F$	$p(a)^F$
$p(a)^T \mid p(b)^T$	$p(a)^T \mid p(b)^T$
$\bot : [a/y] \mid$	$\bot : [a/y] \mid \neg p(z)^T$
	$\mid p(z)^F$
	$\mid \bot : [b/z]$

After we have used up $p(y)^F$ by applying $[a/y]$ in $\mathcal{T}_1^f\bot$, we have to get a new instance $p(z)^F$ via $\mathcal{T}_1^f\forall$.

SOME RIGHTS RESERVED

Termination and Multiplicity in Tableaux

- **Recall:** In $\mathcal{T}_0$, all rules only needed to be applied once.
  $\rightsquigarrow$ $\mathcal{T}_0$ terminates and thus induces a decision procedure for PL^0.
- **Observation 2.14.** *All $\mathcal{T}_1^f$ rules except $\mathcal{T}_1^f\forall$ only need to be applied once.*
- **Example 2.15.** A tableau proof for $(p(a) \vee p(b)) \Rightarrow (\exists.p())$.
- **Definition 2.16.** Let $\mathcal{T}$ be a tableau for A, and a positive occurrence of $\forall x.\mathrm{B}$ in A, then we call the number of applications of $\mathcal{T}_1^f\forall$ to $\forall x.\mathrm{B}$ its multiplicity.
- **Observation 2.17.** *Given a prescribed multiplicity for each positive $\forall$, saturation with $\mathcal{T}_1^f$ terminates.*
- *Proof sketch:* All $\mathcal{T}_1^f$ rules reduce the number of connectives and negative $\forall$ or the multiplicity of positive $\forall$.

SOME RIGHTS RESERVED

Termination and Multiplicity in Tableaux

- **Recall:** In $\mathcal{T}_0$, all rules only needed to be applied once.
  $\rightsquigarrow$ $\mathcal{T}_0$ terminates and thus induces a decision procedure for PL^0.
- **Observation 2.19.** *All $\mathcal{T}_1^f$ rules except $\mathcal{T}_1^f\forall$ only need to be applied once.*
- **Example 2.20.** A tableau proof for $(p(a) \vee p(b)) \Rightarrow (\exists.p())$.
- **Definition 2.21.** Let $\mathcal{T}$ be a tableau for A, and a positive occurrence of $\forall x.\mathrm{B}$ in A, then we call the number of applications of $\mathcal{T}_1^f\forall$ to $\forall x.\mathrm{B}$ its multiplicity.
- **Observation 2.22.** *Given a prescribed multiplicity for each positive $\forall$, saturation with $\mathcal{T}_1^f$ terminates.*
- *Proof sketch:* All $\mathcal{T}_1^f$ rules reduce the number of connectives and negative $\forall$ or the multiplicity of positive $\forall$.
- **Theorem 2.23.** *$\mathcal{T}_1^f$ is only complete with unbounded multiplicities.*
- *Proof sketch:* Replace $p(a) \vee p(b)$ with $p(a_1) \vee \ldots \vee p(a_n)$ in 2.5.

SOME RIGHTS RESERVED

Termination and Multiplicity in Tableaux

- **Recall:** In $\mathcal{T}_0$, all rules only needed to be applied once.
  $\leadsto$ $\mathcal{T}_0$ terminates and thus induces a decision procedure for PL^0.
- **Observation 2.24.** *All $\mathcal{T}_1^f$ rules except $\mathcal{T}_1^f\forall$ only need to be applied once.*
- **Example 2.25.** A tableau proof for $(p(a) \vee p(b)) \Rightarrow (\exists.p())$.
- **Definition 2.26.** Let $\mathcal{T}$ be a tableau for A, and a positive occurrence of $\forall x.B$ in A, then we call the number of applications of $\mathcal{T}_1^f\forall$ to $\forall x.B$ its multiplicity.
- **Observation 2.27.** *Given a prescribed multiplicity for each positive $\forall$, saturation with $\mathcal{T}_1^f$ terminates.*
- *Proof sketch:* All $\mathcal{T}_1^f$ rules reduce the number of connectives and negative $\forall$ or the multiplicity of positive $\forall$.
- **Theorem 2.28.** *$\mathcal{T}_1^f$ is only complete with unbounded multiplicities.*
- *Proof sketch:* Replace $p(a) \vee p(b)$ with $p(a_1) \vee \ldots \vee p(a_n)$ in 2.5.
- **Remark:** Otherwise validity in PL^1 would be decidable.
- **Implementation:** We need an iterative multiplicity deepening process.

SOME RIGHTS RESERVED

Treating $\mathcal{T}_1^f\bot$

- **Recall:** The $\mathcal{T}_1^f\bot$ rule instantiates the whole tableau.
- **Problem:** There may be more than one $\mathcal{T}_1^f\bot$ opportunity on a branch.
- **Example 2.29.** Choosing which matters – this tableau does not close!

$$\begin{array}{c|c}
\multicolumn{2}{c}{(\exists x.(p(a) \wedge p(b) \Rightarrow p()) \wedge (q(b) \Rightarrow q(x)))^F} \\
\multicolumn{2}{c}{((p(a) \wedge p(b) \Rightarrow p()) \wedge (q(b) \Rightarrow q(y)))^F} \\
(p(a) \Rightarrow p(b) \Rightarrow p())^F & (q(b) \Rightarrow q(y))^F \\
p(a)^T & q(b)^T \\
p(b)^T & q(y)^F \\
p(y)^F & \\
\bot : [a/y] &
\end{array}$$

  choosing the other $\mathcal{T}_1^f\bot$ in the left branch allows closure.

- **Idea:** Two ways of systematic proof search in $\mathcal{T}_1^f$:
  - backtracking search over $\mathcal{T}_1^f\bot$ opportunities
  - saturate without $\mathcal{T}_1^f\bot$ and find spanning matings (next slide)

SOME RIGHTS RESERVED

Spanning Matings for $\mathcal{T}_1^f\bot$

- **Observation 2.30.** *$\mathcal{T}_1^f$ without $\mathcal{T}_1^f\bot$ is terminating and confluent for given multiplicities.*
- **Idea:** Saturate without $\mathcal{T}_1^f\bot$ and treat all cuts at the same time (later).
- **Definition 2.31.**
  Let $\mathcal{T}$ be a $\mathcal{T}_1^f$ tableau, then we call a unification problem $\mathcal{E}{:=}\mathrm{A}_1{=^?}\mathrm{B}_1 \wedge \ldots \wedge \mathrm{A}_n{=^?}\mathrm{B}_n$ a mating for $\mathcal{T}$, iff ${\mathrm{A}_i}^{\mathsf{T}}$ and ${\mathrm{B}_i}^{\mathsf{F}}$ occur in the same branch in $\mathcal{T}$.
  We say that $\mathcal{E}$ is a spanning mating, if $\mathcal{E}$ is unifiable and every branch $\mathcal{B}$ of $\mathcal{T}$ contains ${\mathrm{A}_i}^{\mathsf{T}}$ and ${\mathrm{B}_i}^{\mathsf{F}}$ for some i.
- **Theorem 2.32.** *A $\mathcal{T}_1^f$-tableau with a spanning mating induces a closed $\mathcal{T}_1$ tableau.*
- *Proof sketch:* Just apply the unifier of the spanning mating.
- **Idea:** Existence is sufficient, we do not need to compute the unifier.
- **Implementation:** Saturate without $\mathcal{T}_1^f\bot$, backtracking search for spanning matings with $\mathcal{DU}$, adding pairs incrementally.

SOME RIGHTS RESERVED

Spanning Matings for $\mathcal{T}_1^f\bot$

- **Observation 2.33.** *$\mathcal{T}_1^f$ without $\mathcal{T}_1^f\bot$ is terminating and confluent for given multiplicities.*
- **Idea:** Saturate without $\mathcal{T}_1^f\bot$ and treat all cuts at the same time (later).
- **Definition 2.34.**
  Let $\mathcal{T}$ be a $\mathcal{T}_1^f$ tableau, then we call a unification problem $\mathcal{E}{:=}\mathrm{A}_1{=^?}\mathrm{B}_1 \wedge \ldots \wedge \mathrm{A}_n{=^?}\mathrm{B}_n$ a mating for $\mathcal{T}$, iff ${\mathrm{A}_i}^{\mathsf{T}}$ and ${\mathrm{B}_i}^{\mathsf{F}}$ occur in the same branch in $\mathcal{T}$.
  We say that $\mathcal{E}$ is a spanning mating, if $\mathcal{E}$ is unifiable and every branch $\mathcal{B}$ of $\mathcal{T}$ contains ${\mathrm{A}_i}^{\mathsf{T}}$ and ${\mathrm{B}_i}^{\mathsf{F}}$ for some i.
- **Theorem 2.35.** *A $\mathcal{T}_1^f$-tableau with a spanning mating induces a closed $\mathcal{T}_1$ tableau.*
- *Proof sketch:* Just apply the unifier of the spanning mating.
- **Idea:** Existence is sufficient, we do not need to compute the unifier.
- **Implementation:** Saturate without $\mathcal{T}_1^f\bot$, backtracking search for spanning matings with $\mathcal{DU}$, adding pairs incrementally.

SOME RIGHTS RESERVED

8.3 Model Generation with Quantifiers

SOME RIGHTS RESERVED

Model Generation (The *RM* Calculus [Kon04])

- **Idea:** Try to generate domain-minimal (i.e. fewest individuals) models (for NL interpretation)
- **Problem:** Even one function constant makes Herbrand base infinite (solution: leave them out)
- **Definition 3.1.** *RM* adds ground quantifier rules to propositional tableau calculus

$$\frac{(\forall X.A)^{T} \quad c\in\mathcal{H}}{([c/X](A))^{T}}RM{\forall} \qquad \frac{(\forall X.A)^{F} \quad \mathcal{H}=\{a_1,\ldots,a_n\} \quad w\notin\mathcal{H} \text{ new}}{([a_1/X](A))^{F} \;\Big|\; \ldots \;\Big|\; ([a_n/X](A))^{F} \;\Big|\; ([w/X](A))^{F}}$$

  - $RM\exists$ rule introduces new witness constant w to Herbrand base $\mathcal{H}$ of branch
  - Apply $RM\forall$ exhaustively (for new w reapply all $RM\forall$ rules on branch!)

SOME RIGHTS RESERVED

Generating infinite models (Natural Numbers)

- We have to re-apply the $RM\forall$ rule for any new constant
- **Example 3.2.** This leads to the generation of infinite models

$$
\begin{array}{c}
(\forall x.\neg x > x \wedge \ldots)^T \\
N(0)^T \\
(\forall x.N(x) \Rightarrow (\exists y.N(y) \wedge y > x))^T \\
(N(0) \Rightarrow (\exists y.N(y) \wedge y > 0))^T \\
\begin{array}{c|c}
\begin{array}{c} N(0)^F \\ \bot \end{array} &
\begin{array}{c}
(\exists y.N(y) \wedge y > 0)^T \\
\begin{array}{c|c}
\begin{array}{c} 0 > 0^T \\ N(0)^T \\ 0 > 0^F \\ \bot \end{array} &
\begin{array}{c} N(1)^T \\ 1 > 0^T \\ (N(1) \Rightarrow (\exists y.N(y) \wedge y > 1))^T \\
\begin{array}{c|c}
\begin{array}{c} N(1)^F \\ \bot \end{array} &
\begin{array}{c} (\exists y.N(y) \wedge y > 1)^T \\
\begin{array}{c|c|c}
\begin{array}{c} N(0)^T \\ 0 > 1^T \\ \vdots \\ \bot \end{array} &
\begin{array}{c} N(1)^T \\ 1 > 1^T \\ 1 > 1^F \\ \bot \end{array} &
\begin{array}{c} N(2)^T \\ 2 > 1^T \\ \vdots \end{array}
\end{array}
\end{array}
\end{array}
\end{array}
\end{array}
\end{array}
\end{array}
\end{array}
$$

SOME RIGHTS RESERVED

Example: *Peter is a man. No man walks*

without sorts

$$\begin{array}{c} \boxed{\mathrm{man}(\mathrm{peter})} \\ \boxed{\neg(\exists X.\mathrm{man}(X) \wedge \mathrm{walks}(X))} \\ (\exists X.\mathrm{man}(X) \wedge \mathrm{walks}(X))^F \\ (\mathrm{man}(\mathrm{peter}) \wedge \mathrm{walks}(\mathrm{peter}))^F \\ \begin{array}{c|c} \mathrm{man}(\mathrm{peter})^F & \mathrm{walks}(\mathrm{peter})^F \\ \bot & \end{array} \end{array}$$

problem: 1000 women

$\Rightarrow$

1000 closed branches

with sort $\mathbb{M}\mathtt{ale}$

$$\begin{array}{c} \boxed{\mathrm{man}(\mathrm{peter})} \\ \boxed{\neg(\exists X_{\mathbb{M}\mathtt{ale}}.\mathrm{walks}(X))} \\ (\exists X_{\mathbb{M}\mathtt{ale}}.\mathrm{walks}(X))^F \\ \mathrm{walks}(\mathrm{peter})^F \end{array}$$

- Herbrand-model

$$\{\mathrm{man}(\mathrm{peter})^T, \mathrm{walks}(\mathrm{peter})^F\}$$

SOME RIGHTS RESERVED

Anaphor Resolution *A man sleeps. He snores*

$$\boxed{\exists X.\mathrm{sleeps}(X)}$$
$$\mathrm{sleeps}(c^1_{\mathbb{M}\mathrm{an}})^{\mathsf{T}}$$
$$\boxed{\exists Y_{\mathbb{M}\mathrm{an}}.\mathrm{snores}(Y)}$$

$$\mathrm{snores}(c^1_{\mathbb{M}\mathrm{an}})^{\mathsf{T}} \qquad\qquad \mathrm{snores}(c^2_{\mathbb{M}\mathrm{an}})^{\mathsf{T}}$$

minimal | deictic

- In a situation without men (but maybe thousands of women)

SOME RIGHTS RESERVED

Anaphora with World Knowledge

- **Example 3.3.** *Mary is married to Jeff. Her husband is not in town.* (slightly outside $\mathcal{F}_2$) In PL^1: $\mathrm{married}(\mathrm{mary}, \mathrm{jeff})$, and

$$\exists W_{\mathbb{M}\mathtt{ale}}, W'_{\mathbb{F}\mathtt{emale}}.\mathrm{husband}(W, W') \wedge \neg\mathrm{intown}(W)$$

- World knowledge
  - If woman X is married to man Y, then Y is the only husband of X.
  - $\forall X_{\mathbb{F}\mathtt{emale}}, Y_{\mathbb{M}\mathtt{ale}}.\mathrm{married}(X, Y) \Rightarrow \mathrm{husband}(Y, X) \wedge (\forall Z.\mathrm{husband}(Z, X) \Rightarrow (Z = Y))$
- Model generation gives tableau where all open branches contain

$$\{\mathrm{married}(\mathrm{mary}, \mathrm{jeff})^{\mathsf{T}}, \mathrm{husband}(\mathrm{jeff}, \mathrm{mary})^{\mathsf{T}}, \mathrm{intown}(\mathrm{jeff})^{\mathsf{F}}\}$$

- **Differences:** Additional negative facts e.g. $\mathrm{married}(\mathrm{mary}, \mathrm{mary})^{\mathsf{F}}$.

SOME RIGHTS RESERVED

A branch without world knowledge

$$\mathrm{married}(\mathrm{mary}, \mathrm{jeff})^{\mathsf{T}}$$
$$(\exists Z_{\mathbb{M}\mathtt{ale}}, Z'_{\mathbb{F}\mathtt{emale}}.\mathrm{husband}(Z, Z') \wedge \neg\mathrm{intown}(Z))^{\mathsf{T}}$$
$$(\exists Z'.\mathrm{husband}(c^1_{\mathbb{M}\mathtt{ale}}, Z') \wedge \neg\mathrm{intown}(c^1_{\mathbb{M}\mathtt{ale}}))^{\mathsf{T}}$$
$$(\mathrm{husband}(c^1_{\mathbb{M}\mathtt{ale}}, \mathrm{mary}) \wedge \neg\mathrm{intown}(c^1_{\mathbb{M}\mathtt{ale}}))^{\mathsf{T}}$$
$$\mathrm{husband}(c^1_{\mathbb{M}\mathtt{ale}}, \mathrm{mary})^{\mathsf{T}}$$
$$\neg\mathrm{intown}(c^1_{\mathbb{M}\mathtt{ale}})^{\mathsf{T}}$$
$$\mathrm{intown}(c^1_{\mathbb{M}\mathtt{ale}})^{\mathsf{F}}$$

- **Problem:** Bigamy: $c^1_{\mathbb{M}\mathtt{ale}}$ and jeff are husbands of *Mary*!

SOME RIGHTS RESERVED

Chapter 9
Fragment 3: Complex Verb Phrases

SOME RIGHTS RESERVED

9.1 Fragment 3 (Handling Verb Phrases)

SOME RIGHTS RESERVED

New Data (Verb Phrases)

- *Ethel howled and screamed.*
- *Ethel kicked the dog and poisoned the cat.*
- *Fiona liked Jo and loathed Ethel and tolerated Prudence.*
- *Fiona kicked the cat and laughed.*
- *Prudence kicked and scratched Ethel.*
- *Bertie didn't laugh.*
- *Bertie didn't laugh and didn't scream.*
- *Bertie didn't laugh or scream.*
- *Bertie didn't laugh or kick the dog.*
- * *Bertie didn't didn't laugh.*

SOME RIGHTS RESERVED

New Grammar in Fragment 3 (Verb Phrases)

- To account for the syntax we come up with the concept of a verb-phrase (VP)
- **Definition 1.1.** $\mathcal{F}_3$ has the following rules:

S1.	S	$\rightarrow$	NP VP_{+fin}
S2.	S	$\rightarrow$	Sconj$S$
V1.	$VP_{\pm fin}$	$\rightarrow$	$V^i_{\pm fin}$
V2.	$VP_{\pm fin}$	$\rightarrow$	$V^t_{\pm fin}$, NP
V3.	$VP_{\pm fin}$	$\rightarrow$	$VP_{\pm fin}$, conj, $VP_{\pm fin}$
V4.	VP_{+fin}	$\rightarrow$	$BE_{=}$, NP
V5.	VP_{+fin}	$\rightarrow$	BE_{pred}, Adj.
V6.	VP_{+fin}	$\rightarrow$	didn't VP_{-fin}
N1.	NP	$\rightarrow$	N_{pr}
N2.	NP	$\rightarrow$	Pron
N3.	NP	$\rightarrow$	the N

L8.	$BE_{=}$	$\rightarrow$	is
L9.	BE_{pred}	$\rightarrow$	is
L10.	V^i_{-fin}	$\rightarrow$	run, laugh, sing,...
L11.	V^t_{-fin}	$\rightarrow$	read, poison,eat,...

- **Limitations of $\mathcal{F}_3$:**
  - The rule for *didn't* over-generates: * *John didn't didn't run* (need tense for that)
  - $\mathcal{F}_3$ does not allow coordination of transitive verbs (problematic anyways)

SOME RIGHTS RESERVED

Implementing Fragment 3 in GF

- The grammar of Fragment 3 only differs from that of Fragment 2 by
  - Verb phrases: `cat VP; VPf;` infinite and finite verb phrases
  - Verb Form: to distinguish *howl* and *howled* in English

```
param VForm = VInf | VPast;
oper VerbType : Type = {s : VForm => Str };
```

  - English Paradigms to deal with verb forms.

```
mkVP = overload {
  mkVP : (v : VForm => Str) -> VP = \v -> lin VP {s = v};
  mkVP : (v : VForm => Str) -> Str -> VP =
  \v,str -> lin VP {s = table{VInf => v!VInf ++ str; VPast => v!VPast ++ str}};
  mkVP : (v : VForm => Str) -> Str -> (v : VForm => Str) -> VP =
  \v1,str,v2 -> lin VP {s = table{VInf => v1!VInf ++ str ++ v2!VInf;
        VPast => v1!VPast ++ str ++ v2!VPast}};};
mkVPf : Str -> VPf = \str -> lin VPf {s = str};
```


SOME RIGHTS RESERVED

9.2 Dealing with Functions in Logic and Language

SOME RIGHTS RESERVED

Types

- Types are semantic annotations for terms that prevent antinomies
- **Definition 2.1.** Given a set $\mathcal{BT}$ of base types, construct function types: $\alpha \to \beta$ is the type of functions with domain type α and range type β. We call the closure $\mathcal{T}$ of $\mathcal{BT}$ under function types the set of types over $\mathcal{BT}$.
- **Definition 2.2.**
  We will use ι for the type ofindividuals and prop for the type of truth values.
- **Right Associativity:** The type constructor is used as a right-associative operator, i.e. we use $\alpha \to \beta \to \gamma$ as an abbreviation for $\alpha \to \beta \to \gamma$
- **Vector Notation:**
  We will use a kind of vector notation for function types, abbreviating $\alpha_1 \to \ldots \to \alpha_n \to \beta$ with $\overline{\alpha}_n \to \beta$.

SOME RIGHTS RESERVED

Syntactical Categories and Types

- Now, we can assign types to phrasial categories.

Cat	Type	Intuition
S	prop	truth value
NP	ι	individual
N_{pr}	ι	individuals
VP	$\iota \rightarrow$ prop	property
V^i	$\iota \rightarrow$ prop	unary predicate
V^t	$\iota \rightarrow \iota \rightarrow$ prop	binary relation

- For the category conj, we cannot get by with a single type. Depending on where it is used, we need the types
  - prop $\rightarrow$ prop $\rightarrow$ prop for S-coordination in rule $S2$: $S \rightarrow S$, conj, S
  - $\iota \rightarrow$ prop $\rightarrow \iota \rightarrow$ prop $\rightarrow \iota \rightarrow$ prop for VP-coordination in $V3$: $VP \rightarrow VP$, conj, VP.
  - **Note:** Computational Linguistics, often uses a different notation for types: e (entiry) for ι, t (truth value) for prop, and $\langle\alpha,\beta\rangle$ for $\alpha \rightarrow \beta$ (no bracket elision convention).
    So the type for VP-coordination has the form $\langle\langle\iota,[\rangle ling]t,\langle\langle\iota,[\rangle ling]t,\langle\iota,[\rangle ling]t\rangle\rangle$

SOME RIGHTS RESERVED

From Comprehension to β-Conversion

- $\exists F_{\alpha\to\beta}.\forall X_\alpha.FX = A_\beta$ for arbitrary variable X_α and term $A\in wff_\beta(\Sigma_\mathcal{T},\mathcal{V}_\mathcal{T})$ (for each term A and each variable X there is a function $f\in\mathcal{D}_{(\alpha\to\beta)}$, with $f(\varphi(X)) = \mathcal{I}_\varphi(A))$
  - schematic in α, β, X_α and A_β, very inconvenient for deduction
- Transformation in $\mathcal{H}_\Omega$
  - $\exists F_{\alpha\to\beta}.\forall X_\alpha.FX = A_\beta$
  - $\forall X_\alpha.(\lambda X_\alpha.A)X = A_\beta$ $(\exists E)$
    Call the function F whose existence is guaranteed "$(\lambda X_\alpha.A)$"
  - $(\lambda X_\alpha.A)B = [B/X]A_\beta$ $(\forall E)$, in particular for $B\in wff_\alpha(\Sigma_\mathcal{T},\mathcal{V}_\mathcal{T})$.
- **Definition 2.3.** Axiom of β equality: $(\lambda X_\alpha.A)\,B = [B/X](A_\beta)$
- **Idea:** Introduce a new class of formulae (λ-calculus [Chu40])

SOME RIGHTS RESERVED

From Extensionality to η-Conversion

- **Definition 2.4.** Extensionality Axiom: $\forall F_{\alpha \to \beta}.\forall G_{\alpha \to \beta}.(\forall X_\alpha.FX = GX) \Rightarrow F = G$
- **Idea:** Maybe we can get by with a simplified equality schema here as well.
- **Definition 2.5.** We say that A and $\lambda X_\alpha.\mathrm{A}\ X$ are η-equal, (write $\mathrm{A}_{\alpha \to \beta} =_\eta (\lambda X_\alpha.\mathrm{A}\ X)$), iff $X \notin \mathrm{free}(\mathrm{A})$.
- **Theorem 2.6.** *η-equality and Extensionality are equivalent*
- *Proof:* We show that η-equality is special case of extensionality; the converse direction is trivial
  1. Let $\forall X_\alpha.\mathrm{A}X = \mathrm{B}X$, thus $\mathrm{A}X = \mathrm{B}X$ with $\forall E$
  2. $\lambda X_\alpha.\mathrm{A}X = \lambda X_\alpha.\mathrm{B}X$, therefore $\mathrm{A} = \mathrm{B}$ with η
  3. Hence $\forall F_{\alpha \to \beta}.\forall G_{\alpha \to \beta}.(\forall X_\alpha.FX = GX) \Rightarrow F = G$ by twice $\forall I$.
- Axiom of truth values: $\forall F_{\mathrm{prop}}.\forall G_{\mathrm{prop}}.FG \Leftrightarrow F = G$ unsolved.

SOME RIGHTS RESERVED

9.3 Translation for Fragment 3

SOME RIGHTS RESERVED

Translations for Fragment 3

- We will look at the new translation rules (the rest stay the same).

$$T1\colon [X_{\mathrm{NP}}, Y_{VP}]_S \rightsquigarrow VP'(\mathrm{NP}'),\ T3\colon [X_{VP}, Y_{\mathrm{conj}}, Z_{VP}]_{VP} \rightsquigarrow \mathrm{conj}'(VP', VP'),$$
$$T4\colon [X_{V^t}, Y_{\mathrm{NP}}]_{VP} \rightsquigarrow V^{t\prime}(\mathrm{NP}')$$

- The lexical insertion rules will give us two items each for *is*, *and*, and *or*, corresponding to the two types we have given them.

word	type	term	case
BE_{pred}	$\iota \to \mathrm{prop} \to \iota \to \mathrm{prop}$	$\lambda P_{\iota\to\mathrm{prop}}.P$	adjective
BE_{eq}	$\iota \to \iota \to \mathrm{prop}$	$\lambda X_\iota Y_\iota.X = Y$	verb
and	$\mathrm{prop} \to \mathrm{prop} \to \mathrm{prop}$	$\wedge$	S-coord.
and	$\iota \to \mathrm{prop} \to \iota \to \mathrm{prop} \to \iota \to \mathrm{prop}$	$\lambda F_{\iota\to\mathrm{prop}} G_{\iota\to\mathrm{prop}} X_\iota.F(X) \wedge G(X)$	VP-coord.
or	$\mathrm{prop} \to \mathrm{prop} \to \mathrm{prop}$	$\vee$	S-coord.
or	$\iota \to \mathrm{prop} \to \iota \to \mathrm{prop} \to \iota \to \mathrm{prop}$	$\lambda F_{\iota\to\mathrm{prop}} G_{\iota\to\mathrm{prop}} X_\iota.F(X) \vee G(X)$	VP-coord.
didn't	$\iota \to \mathrm{prop} \to \iota \to \mathrm{prop}$	$\lambda P_{\iota\to\mathrm{prop}} X_\iota.\neg P\, X$	

Need to assume the logical connectives as constants of the λ-calculus.

- **Note:** With these definitions, it is easy to restrict ourselves to binary branching in the syntax of the fragment.

SOME RIGHTS RESERVED

Translation Example

▶ **Example 3.1.** *Ethel howled and screamed* to

$$
\begin{array}{ll}
 & (\lambda F_{\iota\to\mathrm{prop}} G_{\iota\to\mathrm{prop}} X_{\iota}.F(X) \wedge G(X))\ \mathrm{howls}\ \mathrm{screams}\ \mathrm{ethel} \\
\to_{\beta} & (\lambda G_{\iota\to\mathrm{prop}} X_{\iota}.\mathrm{howls}(X) \wedge G(X))\ \mathrm{screams}\ \mathrm{ethel} \\
\to_{\beta} & (\lambda X_{\iota}.\mathrm{howls}(X) \wedge \mathrm{screams}(X))\ \mathrm{ethel} \\
\to_{\beta} & \mathrm{howls}(\mathrm{ethel}) \wedge \mathrm{screams}(\mathrm{ethel})
\end{array}
$$

SOME RIGHTS RESERVED

Higher-Order Logic without Quantifiers (HOL_{NQ})

- **Problem:** Need a logic like $\mathrm{PL}^{\mathrm{nq}}$, but with λ-terms to interpret $\mathcal{F}_3$ into.
- **Idea:** Re-use the syntactical framework of $\Lambda^{\rightarrow}$.
- **Definition 3.2.** Let HOL_{NQ} be an instance of $\Lambda^{\rightarrow}$, with $\mathcal{BT} = \{\iota, \mathrm{prop}\}$, $\wedge \in \Sigma_{\mathrm{prop}\rightarrow\mathrm{prop}\rightarrow\mathrm{prop}}$, $\neg \in \Sigma_{\mathrm{prop}\rightarrow\mathrm{prop}}$, and $= \in \Sigma_{\alpha\rightarrow\alpha\rightarrow\mathrm{prop}}$ for all types α.
- **Idea:** To extend this to a semantics for HOL_{NQ}, we only have to say something about the base type prop, and the logical constants $\neg_{\mathrm{prop}\rightarrow\mathrm{prop}}$, $\wedge_{\mathrm{prop}\rightarrow\mathrm{prop}\rightarrow\mathrm{prop}}$, and $=_{\alpha\rightarrow\alpha\rightarrow\mathrm{prop}}$.
- **Definition 3.3.** We define the semantics of HOL_{NQ} by setting
  1. $\mathcal{D}_{\mathrm{prop}} = \{\mathsf{T}, \mathsf{F}\}$; the set of truth values
  2. $\mathcal{I}(\neg) \in \mathcal{D}_{(\mathrm{prop}\rightarrow\mathrm{prop})}$, is the function $\{\mathsf{F} \mapsto \mathsf{T}, \mathsf{T} \mapsto \mathsf{F}\}$
  3. $\mathcal{I}(\wedge) \in \mathcal{D}_{(\mathrm{prop}\rightarrow\mathrm{prop}\rightarrow\mathrm{prop})}$ is the function with $\mathcal{I}(\wedge)@\langle a, b\rangle = \mathsf{T}$, iff $a = \mathsf{T}$ and $b = \mathsf{T}$.
  4. $\mathcal{I}(=) \in \mathcal{D}_{(\alpha\rightarrow\alpha\rightarrow\mathrm{prop})}$ is the identity relation on $\mathcal{D}_{\alpha}$.

SOME RIGHTS RESERVED

9.4 Simply Typed λ-Calculus

SOME RIGHTS RESERVED

Simply typed λ-Calculus (Syntax)

- **Definition 4.1.** Signature $\Sigma_{\mathcal{T}} = \bigcup_{\alpha\in\mathcal{T}}\Sigma_\alpha$ (includes countably infinite signatures Σ_α^{Sk} of Skolem contants).
- $\mathcal{V}_{\mathcal{T}} = \bigcup_{\alpha\in\mathcal{T}}\mathcal{V}_\alpha$, such that $\mathcal{V}_\alpha$ are countably infinite.
- **Definition 4.2.** We call the set $wff_\alpha(\Sigma_{\mathcal{T}},\mathcal{V}_{\mathcal{T}})$ defined by the rules
  - $\mathcal{V}_\alpha \cup \Sigma_\alpha \subseteq wff_\alpha(\Sigma_{\mathcal{T}},\mathcal{V}_{\mathcal{T}})$
  - If $\mathrm{C}\in wff_{\alpha\to\beta}(\Sigma_{\mathcal{T}},\mathcal{V}_{\mathcal{T}})$ and $\mathrm{A}\in wff_\alpha(\Sigma_{\mathcal{T}},\mathcal{V}_{\mathcal{T}})$, then $\mathrm{C}\ \mathrm{A}\in wff_\beta(\Sigma_{\mathcal{T}},\mathcal{V}_{\mathcal{T}})$
  - If $\mathrm{A}\in wff_\alpha(\Sigma_{\mathcal{T}},\mathcal{V}_{\mathcal{T}})$, then $\lambda X_\beta.\mathrm{A}\in wff_{\beta\to\alpha}(\Sigma_{\mathcal{T}},\mathcal{V}_{\mathcal{T}})$

  the set of well typed formulae of type α over the signature $\Sigma_{\mathcal{T}}$ and use $wff_{\mathcal{T}}(\Sigma_{\mathcal{T}},\mathcal{V}_{\mathcal{T}}):=\bigcup_{\alpha\in\mathcal{T}} wff_\alpha(\Sigma_{\mathcal{T}},\mathcal{V}_{\mathcal{T}})$ for the set of all well-typed formulae.
- **Definition 4.3.** We will call all occurrences of the variable X in A bound in $\lambda X.\mathrm{A}$. Variables that are not bound in B are called free in B.
- Substitutions are well typed, i.e. $\sigma(X_\alpha)\in wff_\alpha(\Sigma_{\mathcal{T}},\mathcal{V}_{\mathcal{T}})$ and capture-avoiding.
- **Definition 4.4 (Simply Typed λ-Calculus).** The simply typed λ calculus $\Lambda^{\to}$ over a signature $\Sigma_{\mathcal{T}}$ has the formulae $wff_{\mathcal{T}}(\Sigma_{\mathcal{T}},\mathcal{V}_{\mathcal{T}})$ (they are called λ-terms) and the following equalities:
  - α conversion: $(\lambda X.\mathrm{A}) =_\alpha (\lambda Y.[Y/X](\mathrm{A}))$.
  - β conversion: $(\lambda X.\mathrm{A})\ \mathrm{B}=_\beta[\mathrm{B}/X](\mathrm{A})$.
  - η conversion: $(\lambda X.\mathrm{A}\ X)=_\eta\mathrm{A}$ if $X\notin\mathrm{free}(\mathrm{A})$.

SOME RIGHTS RESERVED

Simply typed λ-Calculus (Notations)

- **Application is left-associative:** We abbreviate $\mathrm{F}\ \mathrm{A}^1\ \mathrm{A}^2\ \ldots\ \mathrm{A}^n$ with $\mathrm{F}(\mathrm{A}^1, \ldots, \mathrm{A}^n)$ eliding the brackets and further with $\mathrm{F}\ \overline{\mathrm{A}^n}$ in a kind of vector notation.
- **Andrews' dot Notation:** $\mathrm{A}\,\mathbf{.}$ stands for a left bracket whose partner is as far right as is consistent with existing brackets; i.e. $\mathrm{A}\,\mathbf{.}\mathrm{B}\ \mathrm{C}$ abbreviates $\mathrm{A}\ (\mathrm{B}\ \mathrm{C})$.
- **Abstraction is right-associative:** We abbreviate $\lambda X^1.\lambda X^2.\cdots\lambda X^n.\mathrm{A}\cdots$ with $\lambda X^1\ldots X^n.\mathrm{A}$ eliding brackets, and further to $\lambda\overline{X^n}.\mathrm{A}$ in a kind of vector notation.
- **Outer brackets:** Finally, we allow ourselves to elide outer brackets where they can be inferred.

SOME RIGHTS RESERVED

$=_{\alpha\beta\eta}$-Equality (Overview)

- **Definition 4.5.** Reduction with $\left\{\begin{array}{ll} =_\beta : & (\lambda X.A)\, B \to_\beta [B/X](A) \\ =_\eta : & (\lambda X.A\, X) \to_\eta A \end{array}\right.$ under

  $=_\alpha : \quad \begin{array}{c} \lambda X.A \\ =_\alpha \\ \lambda Y.[Y/X](A) \end{array}$

- **Theorem 4.6.** *β-reduction is well-typed, terminating and confluent in the presence of α-conversion.*
- **Definition 4.7 (Normal Form).** We call a λ-term A a normal form (in a reduction system $\mathcal{E}$), iff no rule (from $\mathcal{E}$) can be applied to A.
- **Corollary 4.8.** *$=_{\beta\eta}$-reduction yields unique normal forms (up to $=_\alpha$-equivalence).*

SOME RIGHTS RESERVED

Syntactic Parts of λ-Terms

- **Definition 4.9 (Parts of λ-Terms).** We can always write a λ-term in the form $T = \lambda X^1 \ldots X^k.HA^1 \ldots A^n$, where H is not an application. We call
  - H the syntactic head of T
  - $H(A^1, \ldots, A^n)$ the matrix of T, and
  - $\lambda X^1 \ldots X^k.$ (or the sequence $X^1, \ldots, X^k$) the binder of T
- **Definition 4.10.**
  Head reduction always has a unique β redex

$$(\lambda \overline{X^n}.\lambda Y.A(B^2, \ldots, B^n)) \rightarrow_\beta^h (\lambda \overline{X^n}.[B^1/Y](A)(B^2, \ldots, B^n))$$

- **Theorem 4.11.** *The syntactic heads of β-normal forms are constant or variables.*
- **Definition 4.12.** Let A be a λ-term, then the syntactic head of the β-normal form of A is called the head symbol of A and written as $\mathrm{head}(A)$. We call a λ-term a j-projection, iff its head is the j^{th} bound variable.
- **Definition 4.13.** We call a λ-term a η long form, iff its matrix has base type.
- **Definition 4.14.** η Expansion makes η long forms

$$\eta[(\lambda X^1 \ldots X^n.A)] := (\lambda X^1 \ldots X^n.\lambda Y^1 \ldots Y^m.A(Y^1, \ldots, Y^m))$$

- **Definition 4.15.** Long $\beta\eta$ normal form, iff it is β normal and η-long.

SOME RIGHTS RESERVED

Semantics of $\Lambda^{\to}$

- **Definition 4.16.** We call a collection $\mathcal{D}_{\mathcal{T}}:=\{\mathcal{D}_\alpha|\alpha\in\mathcal{T}\}$ a typed collection (of sets) and a collection $f_{\mathcal{T}}\colon\mathcal{D}_{\mathcal{T}}\to\mathcal{E}_{\mathcal{T}}$, a typed function, iff $f_\alpha\colon\mathcal{D}_\alpha\to\mathcal{E}_\alpha$.
- **Definition 4.17.** A typed collection $\mathcal{D}_{\mathcal{T}}$ is called a frame, iff $\mathcal{D}_{(\alpha\to\beta)}\subseteq\mathcal{D}_\alpha\to\mathcal{D}_\beta$
- **Definition 4.18.** Given a frame $\mathcal{D}_{\mathcal{T}}$, and a typed function $\mathcal{I}\colon\Sigma\to\mathcal{D}$, then we call $\mathcal{I}_\varphi\colon \mathit{wff}_{\mathcal{T}}(\Sigma_{\mathcal{T}},\mathcal{V}_{\mathcal{T}})\to\mathcal{D}$ the value function induced by $\mathcal{I}$, iff
  - $\mathcal{I}_\varphi|_{\mathcal{V}_{\mathcal{T}}}=\varphi,\qquad \mathcal{I}_\varphi|_{\Sigma_{\mathcal{T}}}=\mathcal{I}$
  - $\mathcal{I}_\varphi(\mathrm{A}\,\mathrm{B})=\mathcal{I}_\varphi(\mathrm{A})(\mathcal{I}_\varphi(\mathrm{B}))$
  - $\mathcal{I}_\varphi(\lambda X_\alpha.\mathrm{A})$ is that function $f\in\mathcal{D}_{(\alpha\to\beta)}$, such that $f(a)=\mathcal{I}_{(\varphi,[a/X])}(\mathrm{A})$ for all $a\in\mathcal{D}_\alpha$
- **Definition 4.19.** We call a frame $\langle\mathcal{D},\mathcal{I}\rangle$ comprehension closed or a $\Sigma_{\mathcal{T}}$-algebra, iff $\mathcal{I}_\varphi\colon \mathit{wff}_{\mathcal{T}}(\Sigma_{\mathcal{T}},\mathcal{V}_{\mathcal{T}})\to\mathcal{D}$ is total. (every λ-term has a value)

SOME RIGHTS RESERVED

Domain Theory for $\mathcal{F}_3$

- **Observation 1:** We we can reuse the lexicon theories from $\mathcal{F}_1$
- **Observation 2:** We we can even reuse the grammar theory from $\mathcal{F}_1$, if we extend it in the obvious way (Mmt has all we need)

```
theory frag3log_be : ?plnqd =
    include ?frag1log_be ❙
    useVP : pred1 → pred1 ❘= [v] v ❙
    useVPf : pred1 → ι → o ❘= [v,x] v x❙
    and_VP : pred1 → pred1 → pred1 ❘ = [a,b,x] a x ∧ b x ❙
    or_VP : pred1 → pred1 → pred1 ❘ = [a,b,x] a x ∨ b x ❙
    not_VP : pred1 → pred1 ❘ = [a,x] ¬ a x❙
    and_VPf : pred1 → pred1 → pred1 ❘ = [a,b,x] a x ∧ b x ❙
    or_VPf : pred1 → pred1 → pred1 ❘ = [a,b,x] a x ∨ b x ❙
    not_VPf : pred1 → pred1 ❘ = [a,x] ¬ a x❙
❚
```


SOME RIGHTS RESERVED

Chapter 10
Fragment 4: Noun Phrases and Quantification

SOME RIGHTS RESERVED

10.1 Fragment 4

SOME RIGHTS RESERVED

New Data (more Noun Phrases)

- We want to be able to deal with the following sentences (without the "the-NP" trick)
  1. *Peter loved the cat.*, but not * *Peter loved the the cat.*
  2. *John killed a cat with a white tail.*
  3. *Peter chased the gangster in the car.*
  4. *Peter loves every cat.*
  5. *Every man loves a woman.*

SOME RIGHTS RESERVED

New Grammar in Fragment 4 (Common Noun Phrases)

- To account for the syntax we extend the functionality of noun phrases.
- **Definition 1.1.** $\mathcal{F}_4$ adds the rules on the right to $\mathcal{F}_3$ (on the left):

$S1$: $S \rightarrow \mathrm{NP}, VP_{+fin}$, $S2$: $S \rightarrow S, \mathrm{Sconj}$,
$V1$: $VP_{\pm fin} \rightarrow V^{i}_{\pm fin}$, $V2$: $VP_{\pm fin} \rightarrow V^{t}_{\pm fin}, \mathrm{CNP}$,
$V3$: $VP_{\pm fin} \rightarrow VP_{\pm fin}, \mathrm{VPconj}_{\pm fin}$,
$V4$: $VP_{+fin} \rightarrow BE_{=}, \mathrm{NP}$,
$V5$: $VP_{+fin} \rightarrow BE_{pred}, \mathrm{Adj}$,
$V6$: $VP_{+fin} \rightarrow \mathrm{didn't}, VP_{-fin}$, $N1$: $\mathrm{NP} \rightarrow N_{\mathbf{pr}}$,
$N2$: $\mathrm{NP} \rightarrow \mathrm{Pron}$

$N3$: $\mathrm{NP} \rightarrow \mathrm{Det}\mathrm{CNP}$, $N4$: $\mathrm{CNP} \rightarrow N$,
$N5$: $\mathrm{CNP} \rightarrow \mathrm{PP}$, $N6$: $\mathrm{CNP} \rightarrow \mathrm{Adj}$,
$P1$: $\mathrm{PP} \rightarrow P, \mathrm{NP}$, $S3$: $\mathrm{Sconj} \rightarrow \mathrm{conj}, S$,
$V4$: $\mathrm{VPconj}_{\pm fin} \rightarrow \mathrm{conj}, VP_{\pm fin}$,
$L1$: $P \rightarrow \mathrm{with} \mid \mathrm{of} \mid \ldots$

- **Definition 1.2.** A common noun is a noun that describes a type, for example *woman*, or *philosophy* rather than an individual, such as *Amelia Earhart* (proper name).

SOME RIGHTS RESERVED

Implementing Fragment 4 in GF (Grammar)

- The grammar of Fragment 4 only differs from that of Fragment 4 by
  - common noun phrases: `cat CNP; Npr; lincat CNP = NounPhraeType;`
  - prepositional phrases :
    `cat PP; Det; Prep; lincat Npr, Det, Prep, PP = {s: Str}`
  - new grammar rules

```
useDet : Det -> CNP -> NP; -- every book
useNpr : Npr -> NP; -- Bertie
useN : N -> CNP; -- book
usePrep : Prep -> NP -> PP; -- with a book
usePP : PP -> CNP -> CNP; -- teacher with a book
```

  - grammar rules for "special" words that might not belong into the lexicon

Abstract	English
`with_Prep : Prep;`	`with_Prep = mkPrep "with";`
`of_Prep : Prep;`	`of_Prep = mkPrep "of";`
`the_Det : Det;`	`the_Det = mkDet "the";`
`every_Det : Det;`	`every_Det = mkDet "every";`
`a_Det : Det;`	`a_Det = mkDet "a";`

SOME RIGHTS RESERVED

Implementing Fragment 4 in GF (Grammar)

- English Paradigms to deal with (common) noun phrases
  - Another case for `mkNP`

```
mkNP : Str -> (Case => Str) -> NP
    = \prefix,t -> lin NP { s = table { nom => prefix ++ t!nom;
                                        acc => prefix ++ t!acc}};

mkNpr : Str -> Npr = \name -> lin Npr { s = name };
mkDet : Str -> Det = \every -> lin Det { s = every };
mkPrep : Str -> Prep = \p -> lin Prep { s = p };
mkPP : Str -> PP = \s -> lin PP { s = s };
mkCNP = overload {
  mkCNP : Str -> CNP
      = \book -> lin CNP { s = table { nom => book; acc => book } };
  mkCNP : (Case => Str) -> Str -> CNP
      = \t,suffix -> lin CNP { s = table { nom => (t!nom) ++ suffix;
                                           acc => (t!acc) ++ suffix}};};
```

Translation of Determiners and Quantifiers

- **Idea:** We establish the semantics of quantifying determiners by $=_\beta$-expansion.
  1. assume that we are translating into a λ-calculus with quantifiers and that $\forall X.\mathrm{boy}(X) \Rightarrow \mathrm{runs}(X)$ translates *Every boy runs*, and $\exists X.\mathrm{boy}(X) \wedge \mathrm{runs}(X)$ for *Some boy runs*
  2. $\forall\!\!\!\forall := (\lambda P_{\iota\to\mathrm{prop}} Q_{\iota\to\mathrm{prop}}.(\forall.P(X) \Rightarrow Q(X)))$ for *every*. (subset relation)
  3. $\exists\!\!\!\exists := (\lambda P_{\iota\to\mathrm{prop}} Q_{\iota\to\mathrm{prop}}.(\exists.P(X) \wedge Q(X)))$ for *some*. (nonempty intersection)
- **Problem:** Linguistic Quantifiers take two arguments (restriction and scope), logical ones only one! (in logics, restriction is the universal set)
- We cannot treat *the* with regular quantifiers (new logical constant; see below)
- **Definition 1.3.** We translate the to $\tau := (\lambda P_{\iota\to\mathrm{prop}} Q_{\iota\to\mathrm{prop}}.Q\ \iota\ P)$, where ι is a new operator that given a set returns its (unique) member.
- **Example 1.4.** This translates *The pope spoke* to $\tau(\mathrm{pope}, \mathrm{speaks})$, which $=_\beta$-reduces to $\mathrm{speaks}(\iota\ \mathrm{pope})$.

SOME RIGHTS RESERVED

10.2 Inference for Fragment 4

SOME RIGHTS RESERVED

10.2.1 Quantifiers and Equality in Higher-Order Logic

SOME RIGHTS RESERVED

Higher-Order Abstract Syntax

- **Idea:** In $\text{HOL}^{\rightarrow}$, we already have variable binder: λ, use that to treat quantification.
- **Definition 2.1.** We assume logical constants Π^{α} and σ^{α} of type $\alpha \rightarrow \text{prop} \rightarrow \text{prop}$.
  Regain quantifiers as abbreviations:

$$(\forall X_{\alpha}.A) := \Pi^{\alpha}\,(\lambda X_{\alpha}.A) \qquad (\exists X_{\alpha}.A) := \sigma^{\alpha}\,(\lambda X_{\alpha}.A)$$

- **Definition 2.2.** We must fix the semantics of logical constants:
  1. $\mathcal{I}(\Pi^{\alpha})(p) = \mathsf{T}$, iff $p(a) = \mathsf{T}$ for all $a \in \mathcal{D}_{\alpha}$ (i.e. if p is the universal set)
  2. $\mathcal{I}(\sigma^{\alpha})(p) = \mathsf{T}$, iff $p(a) = \mathsf{T}$ for some $a \in \mathcal{D}_{\alpha}$ (i.e. iff p is non-empty)
- With this, we re-obtain the semantics we have given for quantifiers above:

$$\mathcal{I}_{\varphi}(\forall X_{\iota}.A) = \mathcal{I}_{\varphi}(\Pi^{\iota}\,(\lambda X_{\iota}.A)) = \mathcal{I}(\Pi^{\iota})(\mathcal{I}_{\varphi}(\lambda X_{\iota}.A)) = \mathsf{T}$$

iff $\mathcal{I}_{\varphi}(\lambda X_{\iota}.A)(a) = \mathcal{I}_{([a/X],\varphi)}(A) = \mathsf{T}$ for all $a \in \mathcal{D}_{\alpha}$

SOME RIGHTS RESERVED

Equality

- **Definition 2.3 (Leibniz equality).** $\mathrm{Q}^{\alpha}\mathrm{A}_{\alpha}\mathrm{B}_{\alpha} = \forall P_{\alpha\to\mathrm{prop}}.PA \Leftrightarrow PB$ (indiscernability)
- **Note:** $\forall P_{\alpha\to\mathrm{prop}}.PA \Rightarrow PB$ (get the other direction by instantiating P with Q, where $QX \Leftrightarrow (\neg PX)$)
- **Theorem 2.4.** *If $\mathcal{M} = \langle\mathcal{D},\mathcal{I}\rangle$ is a standard model, then $\mathcal{I}_{\varphi}(\mathrm{Q}^{\alpha})$ is the identity relation on $\mathcal{D}_{\alpha}$.*
- **Definition 2.5 (Notation).** We write $\mathrm{A} = \mathrm{B}$ for QAB (A and B are equal, iff there is no property P that can tell them apart.)
- *Proof:*
  1. $\mathcal{I}_{\varphi}(\mathrm{QAB}) = \mathcal{I}_{\varphi}(\forall P.PA \Rightarrow PB) = \mathsf{T}$, iff $\mathcal{I}_{(\varphi,[r/P])}(PA \Rightarrow PB) = \mathsf{T}$ for all $r \in \mathcal{D}_{(\alpha\to\mathrm{prop})}$.
  2. For $\mathrm{A} = \mathrm{B}$ we have $\mathcal{I}_{(\varphi,[r/P])}(PA) = r(\mathcal{I}_{\varphi}(\mathrm{A})) = \mathsf{F}$ or $\mathcal{I}_{(\varphi,[r/P])}(PB) = r(\mathcal{I}_{\varphi}(\mathrm{B})) = \mathsf{T}$.
  3. Thus $\mathcal{I}_{\varphi}(\mathrm{QAB}) = \mathsf{T}$.
  4. Let $\mathcal{I}_{\varphi}(\mathrm{A}) \neq \mathcal{I}_{\varphi}(\mathrm{B})$ and $r = \{\mathcal{I}_{\varphi}(\mathrm{A})\} \in \mathcal{D}_{(\alpha\to\mathrm{prop})}$ (exists in a standard model)
  5. so $r(\mathcal{I}_{\varphi}(\mathrm{A})) = \mathsf{T}$ and $r(\mathcal{I}_{\varphi}(\mathrm{B})) = \mathsf{F}$
  6. $\mathcal{I}_{\varphi}(\mathrm{QAB}) = \mathsf{F}$, as $\mathcal{I}_{(\varphi,[r/P])}(PA \Rightarrow PB) = \mathsf{F}$, since $\mathcal{I}_{(\varphi,[r/P])}(PA) = r(\mathcal{I}_{\varphi}(\mathrm{A})) = \mathsf{T}$ and $\mathcal{I}_{(\varphi,[r/P])}(PB) = r(\mathcal{I}_{\varphi}(\mathrm{B})) = \mathsf{F}$.

SOME RIGHTS RESERVED

Alternative: HOL^∞

- **Definition 2.6.** There is only one logical constant in HOL^∞: $q^\alpha \in \Sigma_{\alpha\to\alpha\to\mathrm{prop}}$ with $\mathcal{I}(q^\alpha)(a,b) = \mathrm{T}$, iff $a = b$.
  We define the rest as below: Definitions (D) and Notations (N)

N	$\mathrm{A}_\alpha = \mathrm{B}_\alpha$	for	$q^\alpha \mathrm{A}_\alpha \mathrm{B}_\alpha$
D	T	for	$q^{\mathrm{prop}} = q^{\mathrm{prop}}$
D	F	for	$\lambda X_{\mathrm{prop}}.T = \lambda X_{\mathrm{prop}}.X_{\mathrm{prop}}$
D	Π^α	for	$q^{\alpha\to\mathrm{prop}}\ (\lambda X_\alpha.T)$
N	$\forall X_\alpha.\mathrm{A}$	for	$\Pi^\alpha\ (\lambda X_\alpha.\mathrm{A})$
D	$\wedge$	for	$\lambda X_{\mathrm{prop}}.\lambda Y_{\mathrm{prop}}.(\lambda G_{\mathrm{prop}\to\mathrm{prop}\to\mathrm{prop}}.G\,T\,T = \lambda G_{\mathrm{prop}\to\mathrm{prop}\to\mathrm{prop}}$
N	$\mathrm{A} \wedge \mathrm{B}$	for	$\wedge\ (\mathrm{A}_{\mathrm{prop}})\ (\mathrm{B}_{\mathrm{prop}})$
D	$\Rightarrow$	for	$\lambda X_{\mathrm{prop}}.\lambda Y_{\mathrm{prop}}.(X = X \wedge Y)$
N	$\mathrm{A} \Rightarrow \mathrm{B}$	for	$\Rightarrow\ (\mathrm{A}_{\mathrm{prop}})\ (\mathrm{B}_{\mathrm{prop}})$
D	$\neg$	for	$q^{\mathrm{prop}}\ F$
D	$\vee$	for	$\lambda X_{\mathrm{prop}}.\lambda Y_{\mathrm{prop}}.\neg(\neg X \wedge \neg Y)$
N	$\mathrm{A} \vee \mathrm{B}$	for	$\vee\ (\mathrm{A}_{\mathrm{prop}})\ (\mathrm{B}_{\mathrm{prop}})$
D	$\exists X_\alpha.\mathrm{A}_{\mathrm{prop}}$	for	$\neg(\forall X_\alpha.\neg \mathrm{A})$
N	$\mathrm{A}_\alpha \neq \mathrm{B}_\alpha$	for	$\neg q^\alpha\ (\mathrm{A}_\alpha)\ (\mathrm{B}_\alpha)$

- yield the intuitive meanings for connectives and quantifiers.

SOME RIGHTS RESERVED

Generalized Quantifiers

- **Problem:** What about *Most boys run.*: linguistically *most* behaves exactly like *every* or *some*.
- **Idea:** *Most boys run* is true just in case the number of boys who run is greater than the number of boys who do not run.

$$\#(\mathcal{I}_\varphi(\text{boy}) \cap \mathcal{I}_\varphi(\text{runs})) > \#(\mathcal{I}_\varphi(\text{boy}) \backslash \mathcal{I}_\varphi(\text{runs}))$$

- **Definition 2.7.** $\#(A) > \#(B)$, iff there is no surjective function from B to A, so we can define

$$most' := (\lambda AB.\neg(\exists F.\forall X.A(X) \wedge \neg B(X) \Rightarrow (\exists.A(Y) \wedge B(Y) \wedge X = F(Y))))$$

SOME RIGHTS RESERVED

Back to *every* and *some* (set characterization)

- We can now give an explicit set characterization of *every* and *some*:
  1. *every* denotes $\{\langle X, Y\rangle | X \subseteq Y\}$
  2. *some* denotes $\{\langle X, Y\rangle | X \cap Y \neq \emptyset\}$
- The denotations can be given in equivalent function terms, as demonstrated above with the denotation of *most*.

SOME RIGHTS RESERVED

10.2.2 Model Generation with Definite Descriptions

SOME RIGHTS RESERVED

Semantics of Definite Descriptions

- **Problem:** We need a semantics for the determiner *the*, as in *the boy runs*
- **Idea (Type):** *the boy* behaves like a proper name (e.g. *Peter*), i.e. has type ι. Applying *the* to a noun (type $\iota \to \mathrm{prop}$) yields ι. So *the* has type $\alpha \to \mathrm{prop} \to \alpha$, i.e. it takes a set as argument.
- **Idea (Semantics):** *the* has the fixed semantics that this function returns the single member of its argument if the argument is a singleton, and is otherwise undefined. (new logical constant)
- **Definition 2.8.** We introduce a new logical constant ι. $\mathcal{I}(\iota)$ is the function $f \in \mathcal{D}_{(\alpha \to \mathrm{prop} \to \alpha)}$, such that $f(s) = a$, iff $s \in \mathcal{D}_{(\alpha \to \mathrm{prop})}$ is the singleton $\{a\}$, and is otherwise undefined. (remember that we can interpret predicates as sets)
- **Axioms for ι:**

$$\forall X_\alpha . X = \iota = X$$
$$\forall P, Q . Q(\iota\ P) \wedge (\forall X, Y . P(X) \wedge P(Y) \Rightarrow X = Y) \Rightarrow (\forall . P(Z) \Rightarrow Q(Z))$$

SOME RIGHTS RESERVED

More Operators and Axioms for HOL$^{\rightarrow}$

- **Definition 2.9.** The unary conditional $w^{\alpha} \in \Sigma_{\mathrm{prop}\rightarrow\alpha\rightarrow\alpha}$
  $w\ (A_{\mathrm{prop}}) B_{\alpha}$ means: "If A, then B".
- **Definition 2.10.** The binary conditional $\mathrm{if}^{\alpha} \in \Sigma_{\mathrm{prop}\rightarrow\alpha\rightarrow\alpha\rightarrow\alpha}$
  $\mathrm{if}\ (A_{\mathrm{prop}})\ (B_{\alpha})\ (C_{\alpha})$ means: "if A, then B else C".
- **Definition 2.11.** The description operator $\iota^{\alpha} \in \Sigma_{\alpha\rightarrow\mathrm{prop}\rightarrow\alpha}$
  if P is a singleton set, then $\iota\ (P_{\alpha\rightarrow\mathrm{prop}})$ is the (unique) element in P.
- **Definition 2.12.** The choice operator $\gamma^{\alpha} \in \Sigma_{\alpha\rightarrow\mathrm{prop}\rightarrow\alpha}$
  if P is non-empty, then $\gamma\ (P_{\alpha\rightarrow\mathrm{prop}})$ is an arbitrary element from P.
- **Definition 2.13 (Axioms for these Operators).**
  - unary conditional: $\forall \varphi_{\mathrm{prop}}. \forall X_{\alpha}. \varphi \Rightarrow w\ \varphi X = X$
  - binary conditional: $\forall \varphi_{\mathrm{prop}}. \forall X_{\alpha}, Y_{\alpha}, Z_{\alpha}. (\varphi \Rightarrow \mathrm{if}\ \varphi\ X\ Y = X) \wedge (\neg\varphi \Rightarrow \mathrm{if}\ \varphi\ Z\ X = X)$
  - description operator $\forall P_{\alpha\rightarrow\mathrm{prop}}. (\exists^{1} X_{\alpha}. PX) \Rightarrow (\forall Y_{\alpha}. PY \Rightarrow \iota\ P = Y)$
  - choice operator $\forall P_{\alpha\rightarrow\mathrm{prop}}. (\exists X_{\alpha}. PX) \Rightarrow (\forall Y_{\alpha}. PY \Rightarrow \gamma\ P = Y)$
- **Idea:** These operators ensure a much larger supply of functions in Henkin models.

SOME RIGHTS RESERVED

More on the Description Operator

- ι is a weak form of the choice operator. (only works on singletons)
- Alternative Axiom of Descriptions: $\forall X_\alpha.\iota^\alpha \;=\; X = X$.
  - use that $\mathcal{I}_{[a/X]}(= \; X) = \{a\}$
  - we only need this for base types $\neq$ prop
  - Define $\iota^{prop} := = \; (\lambda X_{prop}.X)$ or $\iota^{prop} := (\lambda G_{prop \to prop}.G \; T)$ or $\iota^{prop} := = = \; T$
  - $\iota^{(\alpha \to \beta)} := (\lambda H_{\alpha \to \beta \to prop} X_\alpha.\iota^\beta \; (\lambda Z_\beta.(\exists F_{\alpha \to \beta}.H \; F \wedge F \; X = Z)))$

SOME RIGHTS RESERVED

A Model Generation Rule for ι

- **Definition 2.14.**

$$\frac{\begin{array}{c} P(c)^{\mathsf{T}} \\ Q(\iota\ P)^{\alpha} \end{array} \quad \mathcal{H} = \{c, a_1, \ldots, a_n\}}{\begin{array}{c} Q(c)^{\alpha} \\ (P(a_1) \Rightarrow c = a_1)^{\mathsf{T}} \\ \vdots \\ (P(a_n) \Rightarrow c = a_n)^{\mathsf{T}} \end{array}} RM\,\iota$$

- **Intuition:** If we have a member c of P and $Q(\iota\ P)$ is defined (it has truth value $\alpha \in \{\mathsf{T}, \mathsf{F}\}$), then P must be a singleton (i.e. all other members X of P are identical to c) and Q must hold on c. So the rule $RM\,\iota$ forces it to be by making all other members of P equal to c.

SOME RIGHTS RESERVED

Mary owned a lousy computer. The hard drive crashed.

$$(\forall X.\mathrm{computer}(X) \Rightarrow (\exists Y.\mathrm{harddrive}(Y) \wedge \mathrm{partof}(Y,X)))^{\mathsf{T}}$$

$$\boxed{(\exists X.\mathrm{computer}(X) \wedge \mathrm{lousy}(X) \wedge \mathrm{own}(\mathrm{mary},X))^{\mathsf{T}}}$$

$$\mathrm{computer}(c)^{\mathsf{T}}$$

$$\mathrm{lousy}(c)^{\mathsf{T}}$$

$$\mathrm{own}(\mathrm{mary},c)^{\mathsf{T}}$$

$$\begin{array}{c|c}
\mathrm{harddrive}(c)^{\mathsf{T}} & \mathrm{harddrive}(d)^{\mathsf{T}} \\
\mathrm{partof}(c,c)^{\mathsf{T}} & \mathrm{partof}(d,c)^{\mathsf{T}} \\
\vdots & \boxed{\mathrm{crashes}(\iota\ \mathrm{harddrive})^{\mathsf{T}}} \\
\bot & \mathrm{crashes}(d)^{\mathsf{T}} \\
 & (\mathrm{harddrive}(\mathrm{mary}) \Rightarrow \mathrm{mary} = d)^{\mathsf{T}} \\
 & (\mathrm{harddrive}(c) \Rightarrow c = d)^{\mathsf{T}}
\end{array}$$

SOME RIGHTS RESERVED

Another Example *The dog barks*

- In a situation, where there are two dogs: Fido and Chester

$$
\begin{array}{c|c}
\multicolumn{2}{c}{\mathrm{dog}(\mathrm{fido})^{\mathsf{T}}} \\
\multicolumn{2}{c}{\mathrm{dog}(\mathrm{chester})^{\mathsf{T}}} \\
\multicolumn{2}{c}{\boxed{\mathrm{bark}(\iota\ \mathrm{dog})}} \\
\multicolumn{2}{c}{\mathrm{bark}(\mathrm{fido})^{\mathsf{T}}} \\
\multicolumn{2}{c}{(\mathrm{dog}(\mathrm{chester}) \Rightarrow \mathrm{chester} = \mathrm{fido})^{\mathsf{T}}} \\
\mathrm{dog}(\mathrm{chester})^{\mathsf{F}} & \mathrm{chester} = \mathrm{fido}^{\mathsf{T}} \\
\bot &
\end{array} \tag{1}
$$

- Note that none of our rules allows us to close the right branch, since we do not know that Fido and Chester are distinct. Indeed, they could be the same dog (with two different names). But we can eliminate this possibility by adopting a new assumption.

SOME RIGHTS RESERVED

10.2.3 Model Generation with Unique Name Assumptions

SOME RIGHTS RESERVED

Model Generation with Unique Name Assumption (UNA)

- **Problem:** Names are unique usually in natural language
- **Definition 2.15.** The unique name assumption (UNA) makes the assumption that names are unique (in the respective context)
- **Idea:** Add background knowledge of the form $n = m^{\mathsf{F}}$ (n and m names)
- **Better Idea:** Build UNA into the calculus: partition the Herbrand base $\mathcal{H} = \mathcal{U} \cup \mathcal{W}$ into subsets $\mathcal{U}$ for constants with a UNA, and $\mathcal{W}$ without. (treat them differently)
- **Definition 2.16 (Model Generation with UNA).** We add the following two rules to the RM calculus to deal with the unique name assumption.

$$\frac{\begin{array}{c} a = b^{\mathsf{T}} \\ \mathsf{A}^{\alpha} \end{array} \quad a \in \mathcal{W} \quad b \in \mathcal{H}}{([b/a](\mathsf{A}))^{\alpha}} \, RM\,\text{subst} \qquad \frac{a = b^{\mathsf{T}} \quad a, b \in \mathcal{U}}{\bot} \, RM\,\text{una}$$

SOME RIGHTS RESERVED

Solving a Crime with Unique Names

- **Example 2.17.** Tony has observed (at most) two people. Tony observed a murderer that had black hair. It turns out that Bill and Bob were the two people Tony observed. Bill is blond, and Bob has black hair. (Who was the murderer.)
  Let $\mathcal{U} = \{\mathrm{Bill}, \mathrm{Bob}\}$ and $\mathcal{W} = \{\mathrm{murderer}\}$:

$$
\begin{array}{c}
(\forall z.\mathrm{observes}(\mathrm{Tony}, z) \Rightarrow (z = \mathrm{Bill} \lor z = \mathrm{Bob}))^{\mathsf{T}} \\
\mathrm{observes}(\mathrm{Tony}, \mathrm{Bill})^{\mathsf{T}} \\
\mathrm{observes}(\mathrm{Tony}, \mathrm{Bob})^{\mathsf{T}} \\
\mathrm{observes}(\mathrm{Tony}, \mathrm{murderer})^{\mathsf{T}} \\
\mathrm{black\_hair}(\mathrm{murderer})^{\mathsf{T}} \\
\neg\mathrm{black\_hair}(\mathrm{Bill})^{\mathsf{T}} \\
\mathrm{black\_hair}(\mathrm{Bill})^{\mathsf{F}} \\
\mathrm{black\_hair}(\mathrm{Bob})^{\mathsf{T}} \\
(\mathrm{observes}(\mathrm{Tony}, \mathrm{murderer}) \Rightarrow (\mathrm{murderer} = \mathrm{Bill} \lor \mathrm{murderer} = \mathrm{Bob}))^{\mathsf{T}} \\
(\mathrm{murderer} = \mathrm{Bill} \lor \mathrm{murderer} = \mathrm{Bob})^{\mathsf{T}} \\
\begin{array}{c|c}
\mathrm{murderer} = \mathrm{Bill}^{\mathsf{T}} & \mathrm{murderer} = \mathrm{Bob}^{\mathsf{T}} \\
\mathrm{black\_hair}(\mathrm{Bill})^{\mathsf{T}} & \\
\bot &
\end{array}
\end{array}
$$

SOME RIGHTS RESERVED

Rabbits [Gardent & Konrad '99]

- Interpret "the" as $\lambda PQ.Q\iota\, P \wedge \mathrm{uniq}(P)$
  where $\mathrm{uniq}{:=}(\lambda P.(\exists X.P(X) \wedge (\forall Y.P(Y) \Rightarrow X = Y)))$
  and $\forall\!\!\!\forall{:=}(\lambda PQ.(\forall X.P(X) \Rightarrow Q(X)))$.
- "the rabbit is cute", has logical form $\mathrm{uniq}(\mathrm{rabbit}) \wedge (\mathrm{rabbit} \subseteq \mathrm{cute})$.
- *RM* generates $\{\ldots, \mathrm{rabbit}(c), \mathrm{cute}(c)\}$ in situations with at most 1 rabbit.
  (special *RM* $\exists$ rule yields identification and accommodation (c^{new}))

+ At last an approach that takes world knowledge into account!

– tractable only for toy discourses/ontologies
  The world cup final was watched on TV by 7 million people.
  A rabbit is in the garden.
  $\forall X.\mathrm{human}(x)\exists Y.\mathrm{human}(X) \wedge \mathrm{father}(X, Y) \qquad \forall X, Y.\mathrm{father}(X, Y) \Rightarrow X \neq Y$

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG
SOME RIGHTS RESERVED

More than one Rabbit

- **Problem:** What about two rabbits?
  Bugs and Bunny are rabbits. Bugs is in the hat. Jon removes the rabbit from the hat.
- **Idea: Uniqueness under Scope [Gardent & Konrad '99]:**
  - refine *the* to $\lambda PRQ.\mathrm{uniq}(P \cap R \wedge \forall\!\!\!\forall(P \cap R, Q))$
    where R is an "identifying property" (identified from the context and passed as an arbument to *the*)
  - here R is "being in the hat" (by world knowledge about removing)
  - makes Bugs unique (in $P \cap R$) and the discourse acceptable.
- **Idea:** [Hobbs & Stickel&…]:
  - use generic relation rel for "relatedness to context" for P^2.
  - ?? Is there a general theory of relatedness?

SOME RIGHTS RESERVED

10.3 Davidsonian Semantics: Treating Verb Modifiers

SOME RIGHTS RESERVED

Event semantics: Davidsonian Systems

- **Problem:** How to deal with argument structure of (action verbs) and their modifiers
  - *John killed a cat with a hammer.*
- **Idea:** Just add an argument to kills for express the means
- **Problem:** But there may be more modifiers
  1. *Peter killed the cat in the bathroom with a hammer.*
  2. *Peter killed the cat in the bathroom with a hammer at midnight.*

  So we would need a lot of different predicates for the verb *killed*. (impractical)
- **Definition 3.1.** In event semantics we extend the argument structure of (action) verbs contains a 'hidden' argument, the event argument, then treat modifiers as predicates (often called roles) over events [Dav67a].
- **Example 3.2.**
  1. $\exists e.\exists x, y.\mathrm{bathroom}(x) \wedge \mathrm{hammer}(y) \wedge \mathrm{kill}(e, \mathrm{peter}, \iota\,\mathrm{cat}) \wedge \mathrm{in}(e, x) \wedge \mathrm{with}(e, y)$
  2. $\exists e.\exists x, y.\mathrm{bathroom}(x) \wedge \mathrm{hammer}(y) \wedge \mathrm{kill}(e, \mathrm{peter}, \iota\,\mathrm{cat}) \wedge \mathrm{in}(e, x) \wedge \mathrm{with}(e, y) \wedge \mathrm{at}(e, 24:00)$

SOME RIGHTS RESERVED

Event semantics: Neo-Davidsonian Systems

- **Idea:** Take apart the Davidsonian predicates even further, add event participants via thematic roles (from [Par90]).
- **Definition 3.3.** Neo-Davisonian semantics extends event semantics by adding two standardized roles: the agent $\mathrm{ag}(e,s)$ and the patient $\mathrm{pat}(e,o)$ for the subject s and direct object d of the event e.
- **Example 3.4.** Translate *John killed a cat with a hammer.* as $\exists e.\exists x.\mathrm{hammer}(x) \wedge \mathrm{killing}(e) \wedge \mathrm{ag}(e,\mathrm{peter}) \wedge \mathrm{pat}(e,\iota\,\mathrm{cat}) \wedge \mathrm{with}(e,x)$
- **Further Elaboration:** Events can be broken down into sub-events and modifiers can predicate over sub-events.
- **Example 3.5.** The "process" of climbing Mt. Everest starts with the "event" of (optimistically) leaving the base camp and culminates with the "achievement" of reaching the summit (being completely exhausted).
- **Note:** This system can get by without functions, and only needs unary and binary predicates. (well-suited for model generation)

SOME RIGHTS RESERVED

Event types and properties of events

- **Example 3.6 (Problem).** Some (temporal) modifiers are incompatible with some events, e.g. in English progressive:
  1. *He is eating a sandwich* and *He is pushing the cart.*, but not
  2. * *He is being tall.* or * *He is finding a coin.*
- **Definition 3.7 (Types of Events).** There are different types of events that go with different temporal modifiers. [Ven57] distinguishes
  1. states: e.g. *know the answer*, *stand in the corner*
  2. processes: e.g. *run*, *eat*, *eat apples*, *eat soup*
  3. accomplishments: e.g. *run a mile*, *eat an apple*, and
  4. achievements: e.g. *reach the summit*
- **Observations:**
  1. processes and accomplishments appear in the progressive (1),
  2. states and achievements do not (2).
- **Definition 3.8.** The in test
  1. states and activities, but not accomplishments and achievements are compatible with *for*-adverbials
  2. whereas the opposite holds for in-adverbials (5).
- **Example 3.9.**
  1. *run a mile in an hour* vs. * *run a mile for an hour*, but
  2. * *reach the summit for an hour* vs *reach the summit in an hour*

SOME RIGHTS RESERVED

Chapter 11
Davidsonian Semantics: Treating Verb Modifiers

SOME RIGHTS RESERVED

Event semantics: Davidsonian Systems

- **Problem:** How to deal with argument structure of (action verbs) and their modifiers
  - *John killed a cat with a hammer.*
- **Idea:** Just add an argument to kills for express the means
- **Problem:** But there may be more modifiers
  1. *Peter killed the cat in the bathroom with a hammer.*
  2. *Peter killed the cat in the bathroom with a hammer at midnight.*

  So we would need a lot of different predicates for the verb *killed*. (impractical)
- **Definition 0.1.** In event semantics we extend the argument structure of (action) verbs contains a 'hidden' argument, the event argument, then treat modifiers as predicates (often called roles) over events [Dav67a].
- **Example 0.2.**
  1. $\exists e.\exists x, y.\mathrm{bathroom}(x) \wedge \mathrm{hammer}(y) \wedge \mathrm{kill}(e, \mathrm{peter}, \iota\ \mathrm{cat}) \wedge \mathrm{in}(e, x) \wedge \mathrm{with}(e, y)$
  2. $\exists e.\exists x, y.\mathrm{bathroom}(x) \wedge \mathrm{hammer}(y) \wedge \mathrm{kill}(e, \mathrm{peter}, \iota\ \mathrm{cat}) \wedge \mathrm{in}(e, x) \wedge \mathrm{with}(e, y) \wedge \mathrm{at}(e, 24:00)$

SOME RIGHTS RESERVED

Event semantics: Neo-Davidsonian Systems

- **Idea:** Take apart the Davidsonian predicates even further, add event participants via thematic roles (from [Par90]).
- **Definition 0.3.** Neo-Davisonian semantics extends event semantics by adding two standardized roles: the agent $\mathrm{ag}(e, s)$ and the patient $\mathrm{pat}(e, o)$ for the subject s and direct object d of the event e.
- **Example 0.4.** Translate *John killed a cat with a hammer.* as $\exists e.\exists x.\mathrm{hammer}(x) \wedge \mathrm{killing}(e) \wedge \mathrm{ag}(e, \mathrm{peter}) \wedge \mathrm{pat}(e, \iota\, \mathrm{cat}) \wedge \mathrm{with}(e, x)$
- **Further Elaboration:** Events can be broken down into sub-events and modifiers can predicate over sub-events.
- **Example 0.5.** The "process" of climbing Mt. Everest starts with the "event" of (optimistically) leaving the base camp and culminates with the "achievement" of reaching the summit (being completely exhausted).
- **Note:** This system can get by without functions, and only needs unary and binary predicates. (well-suited for model generation)

SOME RIGHTS RESERVED

Event types and properties of events

- **Example 0.6 (Problem).** Some (temporal) modifiers are incompatible with some events, e.g. in English progressive:
  1. *He is eating a sandwich* and *He is pushing the cart.*, but not
  2. * *He is being tall.* or * *He is finding a coin.*
- **Definition 0.7 (Types of Events).** There are different types of events that go with different temporal modifiers. [Ven57] distinguishes
  1. states: e.g. *know the answer*, *stand in the corner*
  2. processes: e.g. *run*, *eat*, *eat apples*, *eat soup*
  3. accomplishments: e.g. *run a mile*, *eat an apple*, and
  4. achievements: e.g. *reach the summit*
- **Observations:**
  1. processes and accomplishments appear in the progressive (1),
  2. states and achievements do not (2).
- **Definition 0.8.** The in test
  1. states and activities, but not accomplishments and achievements are compatible with *for*-adverbials
  2. whereas the opposite holds for in-adverbials (5).
- **Example 0.9.**
  1. *run a mile in an hour* vs. * *run a mile for an hour*, but
  2. * *reach the summit for an hour* vs *reach the summit in an hour*

SOME RIGHTS RESERVED

Part 2
Topics in Semantics

SOME RIGHTS RESERVED

Chapter 12
Dynamic Approaches to NL Semantics

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG
SOME RIGHTS RESERVED

12.1 Discourse Representation Theory

SOME RIGHTS RESERVED

Anaphora and Indefinites revisited (Data)

- **Observation:** We have concentrated on single sentences so far; let's do better.
- **Definition 1.1.** A discourse is a a unit of natural language longer than a single sentence.
- **New Data:** discourses interact with anaphora.:
  - *Peter^1 is sleeping. He$_1$ is snoring.* (normal anaphoric reference)
  - *A man^1 is sleeping. He$_1$ is snoring.* (Scope of existential?)
  - *Peter has a car^1. It$_1$ is parked outside.* (even if this worked)
  - * *Peter has no car^1. It$_1$ is parked outside.* (what about negation?)
  - *There is a book^1 that Peter does not own. It$_1$ is a novel.* (OK)
  - * *Peter does not own every book^1. It$_1$ is a novel.* (equivalent in PL^1)
  - *If a farmer^1 owns a donkey$_2$, he$_1$ beats it$_2$.* (even inside sentences)

SOME RIGHTS RESERVED

Dynamic Effects in Natural Language

- **Problem:** E.g. Quantifier Scope
  - * *A man sleeps. He snores.*
  - $(\exists X.\mathrm{man}(X) \wedge \mathrm{sleeps}(X)) \wedge \mathrm{snores}(X)$
  - X is bound in the first conjunct, and free in the second.
- **Problem:** donkey sentence: *If a farmer owns a donkey, he beats it.*
  $\forall X, Y.\mathrm{farmer}(X) \wedge \mathrm{donkey}(Y) \wedge \mathrm{own}(X, Y) \Rightarrow \mathrm{beat}(X, Y)$
- **Ideas:**
  - Composition of sentences by conjunction inside the scope of existential quantifiers (non-compositional, ...)
  - Extend the scope of quantifiers dynamically (DPL)
  - Replace existential quantifiers by something else (DRT)

SOME RIGHTS RESERVED

Discourse Representation Theory (DRT)

- **Definition 1.2.** Discourse Representation Theory (DRT) is a logical system, which uses discourse referents to model quantification and pronouns. DRT formulae are called discourse representation structure (DRS); these introduce a set of discourse referents and specify their meaning by conditions:
  - atomic propositions,
  - dynamic negations $\neg D$,
  - dynamic implications $D \Rrightarrow E$, and
  - dynamic disjunctions $D \veebar E$.
- Discourse referents e.g. in *A student owns a book.*
  - are kept in a dynamic context ($\leadsto$ accessibility)
  - are declared e.g. in indefinite nominals
  - specified in conditions via predicates

$$\begin{array}{|l|}\hline X, Y \\ \hline \mathrm{student}(X) \\ \mathrm{book}(Y) \\ \mathrm{own}(X, Y) \\ \hline \end{array}$$

- Discourse representation structures (DRS)

A student owns a book. He reads it.

$$\begin{array}{|l|}\hline X, Y, R, S \\ \hline \mathrm{student}(X) \\ \mathrm{book}(Y) \\ \mathrm{own}(X, Y) \\ \mathrm{read}(R, S) \\ X = R \\ Y = S \\ \hline \end{array}$$

If a farmer owns a donkey, he beats it.

$$\begin{array}{|c|}\hline \\ \hline \begin{array}{|l|}\hline X, Y \\ \hline \mathrm{farmer}(X) \\ \mathrm{donkey}(Y) \\ \mathrm{own}(X, Y) \\ \hline \end{array} \Rrightarrow \begin{array}{|l|}\hline \\ \hline \mathrm{beat}(X, Y) \\ \hline \end{array} \\ \hline \end{array}$$

SOME RIGHTS RESERVED

Discourse DRS Construction

- **Problem:** How do we construct DRSes for multi-sentence discourses?
- **Solution:** We construct sentence DRSes individually and merge them (DRSes and conditions separately)
- **Example 1.3.** A three-sentence discourse. (not quite Shakespeare)

Mary sees John.	*John kills a cat.*	*Mary calls a cop.*	merge
$\begin{array}{\|l\|}\hline \\ \hline \mathrm{see}(\mathrm{mary},\mathrm{john}) \\ \hline \end{array}$	$\begin{array}{\|l\|}\hline U \\ \hline \mathrm{cat}(U) \\ \mathrm{kills}(\mathrm{john},U) \\ \hline \end{array}$	$\begin{array}{\|l\|}\hline V \\ \hline \mathrm{policeman}(V) \\ \mathrm{calls}(\mathrm{mary},V) \\ \hline \end{array}$	$\begin{array}{\|l\|}\hline U,V \\ \hline \mathrm{see}(\mathrm{mary},\mathrm{john}) \\ \mathrm{cat}(U) \\ \mathrm{kills}(\mathrm{john},U) \\ \mathrm{policeman}(V) \\ \mathrm{calls}(\mathrm{mary},V) \\ \hline \end{array}$

- Sentence composition via the DRT Merge Operator $\otimes$. (acts on DRSes)

SOME RIGHTS RESERVED

Anaphor Resolution in DRT

- **Problem:** How do we resolve anaphora in DRT?
- **Solution:** Two phases
  - translate pronouns into discourse referents (semantics construction)
  - identify (equate) coreferring discourse referents, (maybe) simplify (semantic/pragmatic analysis)
- **Example 1.4.** *A student owns a book. He reads it.*

A student^1 *owns a book*^{2}*. He*$_1$ *reads it*$_2$

X, Y, R, S
student(X) book(Y) read(R, S)

resolution

X, Y, R, S
student(X) book(Y) read(R, S) $X = R$ $Y = S$

simplify

X, Y
student(X) book(Y) read(X, Y)

SOME RIGHTS RESERVED

DRT (Syntax)

- **Definition 1.5.** Given a set $\mathcal{DR}$ of discourse referents, discourse representation structure (DRSes) are given by the following grammar:

$$\begin{array}{ll} \text{conditions} & \mathcal{C} ::= p(a_1, \ldots, a_n) \mid \mathcal{C}_1 \wedge \mathcal{C}_2 \mid \neg\!\!\neg \mathcal{D} \mid \mathcal{D}_1 \Cup \mathcal{D}_2 \mid \mathcal{D}_1 \Rrightarrow \mathcal{D}_2 \\ \text{DRSes} & \mathcal{D} ::= \delta U^1, \ldots, U^n.\mathcal{C} \mid \mathcal{D}_1 \otimes \mathcal{D}_2 \mid \mathcal{D}_1 \,;;\, \mathcal{D}_2 \end{array}$$

- $\otimes$ and ;; are for sentence composition ($\otimes$ from DRT, ;; from DPL)
- **Example 1.6.** $\delta U, V.\text{farmer}(U) \wedge \text{donkey}(V) \wedge \text{own}(U, V) \wedge \text{beat}(U, V)$
- **Definition 1.7.** The meaning of $\otimes$ and ;; is given operationally by $=_\tau$ Equality:

$$\begin{array}{rcl} \delta\mathcal{X}.\mathcal{C}_1 \otimes \delta\mathcal{Y}.\mathcal{C}_2 & \rightarrow_\tau & \delta\mathcal{X}, \mathcal{Y}.\mathcal{C}_1 \wedge \mathcal{C}_2 \\ \delta\mathcal{X}.\mathcal{C}_1 \,;;\, \delta\mathcal{Y}.\mathcal{C}_2 & \rightarrow_\tau & \delta\mathcal{X}, \mathcal{Y}.\mathcal{C}_1 \wedge \mathcal{C}_2 \end{array}$$

- Discourse referents used instead of bound variables. (specify scoping independently of logic)
- **Idea:** Semantics inherited from first-order logic by a translation mapping.

SOME RIGHTS RESERVED

Sub DRSes and Accessibility

- **Problem:** How can we formally define accessibility. (to make predictions)
- **Idea:** Make use of the structural properties of DRT.
- **Definition 1.8.** A referent is accessible in all sub DRS of the declaring DRS.
  - If $\mathcal{D} = \delta U^1, \ldots, U^n.\mathcal{C}$, then any sub DRS of $\mathcal{C}$ is a sub DRS of $\mathcal{D}$.
  - If $\mathcal{D} = \mathcal{D}^1 \otimes \mathcal{D}^2$, then $\mathcal{D}^1$ is a sub DRS of $\mathcal{D}^2$ and vice versa.
  - If $\mathcal{D} = \mathcal{D}^1 \,;;\, \mathcal{D}^2$, then $\mathcal{D}^2$ is a sub DRS of $\mathcal{D}^1$.
  - If $\mathcal{C}$ is of the form $\mathcal{C}^1 \wedge \mathcal{C}^2$, or $\neg\mathcal{D}$, or $\mathcal{D}^1 \vee\!\!\vee \mathcal{D}^2$, or $\mathcal{D}^1 \Rrightarrow \mathcal{D}^2$, then any sub DRS of the $\mathcal{C}^i$, and the $\mathcal{D}^i$ is a sub DRS of $\mathcal{C}$.
  - If $\mathcal{D} = \mathcal{D}^1 \Rrightarrow \mathcal{D}^2$, then $\mathcal{D}^2$ is a sub DRS of $\mathcal{D}^1$
- **Definition 1.9 (Dynamic Potential).** (which referents can be picked up?) A referent U is in the dynamic potential of a DRS $\mathcal{D}$, iff it is accessible in

$$\mathcal{D} \otimes \begin{array}{|c|}\hline \\ \hline p(U) \\ \hline\end{array}$$

- **Definition 1.10.** We call a DRS static, iff the dynamic potential is empty, and dynamic, if it is not.

Sub DRSes and Accessibility

- **Observation:** Accessibility gives DRSes the flavor of binding structures. (with non-standard scoping!)
- **Idea:** Apply the usual binding heuristics to DRT, e.g.
  - reject DRSes with unbound discourse referents.
- **Questions:** if view of discourse referents as "nonstandard bound variables"
  - what about renaming referents?

SOME RIGHTS RESERVED

Translation from DRT to FOL

- **Definition 1.11.** For $=_\tau$-normal (fully merged) DRSes use the translation $\overline{\cdot}$:

$$\begin{array}{rcl}
\overline{\delta U^1, \ldots, U^n.\mathcal{C}} & = & \exists U^1, \ldots, U^n.\overline{\mathcal{C}} \\
\overline{\neg\mathcal{D}} & = & \neg\overline{\mathcal{D}} \\
\overline{\mathcal{D} \mathbb{V} \mathcal{E}} & = & \overline{\mathcal{D}} \vee \overline{\mathcal{E}} \\
\overline{\mathcal{D} \wedge \mathcal{E}} & = & \overline{\mathcal{D}} \wedge \overline{\mathcal{E}} \\
\overline{(\delta U^1, \ldots, U^n.\mathcal{C}_1) \Rrightarrow (\delta V^1, \ldots, V^n.\mathcal{C}_2)} & = & \forall U^1, \ldots, U^n.\overline{\mathcal{C}_1} \Rightarrow (\exists V^1, \ldots, V^n.\overline{\mathcal{C}_2})
\end{array}$$

- **Example 1.12.** $\overline{\boxed{\begin{array}{l} X, Y \\ \hline \text{student}(X) \\ \text{book}(Y) \\ \text{own}(X, Y) \end{array}}} = \exists X.\exists Y.\text{student}(X) \wedge \text{book}(Y) \wedge \text{own}(X, Y)$.

- **Example 1.13.**

$$\overline{(\delta U, V.\text{farmer}(U) \wedge \text{donkey}(V) \wedge \text{own}(U, V)) \Rrightarrow (\delta W.\text{stick}(W) \wedge \text{beatwith}(U, V, W))}$$
$$= \forall X, Y.\text{farmer}(X) \wedge \text{donkey}(X) \wedge \text{own}(X, Y) \Rightarrow (\exists.\text{stick}(Z) \wedge \text{beatwith}(Z, X, Y))$$

- **Consequence:** Validity of DRSes can be checked by translation.
- **Question:** Why not use first-order logic directly?
- **Answer:** Only translate at the end of a discourse (translation closes all dynamic contexts: frequent re-translation).

FAU FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG
SOME RIGHTS RESERVED

Properties of Dynamic Scope

- **Idea:** Test DRT on the data above for the dynamic phenomena
- **Example 1.14 (Negation Closes Dynamic Potential).**
  Peter has no[1] car. * *It$_1$ is parked outside.*

$$\begin{array}{|c|}\hline \\ \hline \neg\begin{array}{|l|}\hline U \\ \hline \mathrm{acar}(U) \\ \mathrm{own}(\mathrm{peter}, U) \\ \hline\end{array} \\ \hline\end{array} \otimes \begin{array}{|l|}\hline \\ \hline \mathrm{parked}(U) \\ \hline\end{array} \qquad \neg(\exists U.\mathrm{acar}(U) \wedge \mathrm{own}(\mathrm{peter}, U))\ldots$$

- **Example 1.15 (Universal Quantification is Static).**
  Peter does not own every book[1]. * *It$_1$ is a novel.*

$$\begin{array}{|c|}\hline \\ \hline \neg\begin{array}{|c|}\hline \\ \hline \begin{array}{|l|}\hline U \\ \hline \mathrm{book}(U) \\ \hline\end{array} \Rrightarrow \begin{array}{|l|}\hline \\ \hline \mathrm{own}(\mathrm{peter}, U) \\ \hline\end{array} \\ \hline\end{array} \\ \hline\end{array} \otimes \begin{array}{|l|}\hline \\ \hline \mathrm{novel}(U) \\ \hline\end{array}$$

$\neg(\forall U.\mathrm{book}(U) \Rightarrow \mathrm{own}(\mathrm{peter}, U))\ldots$

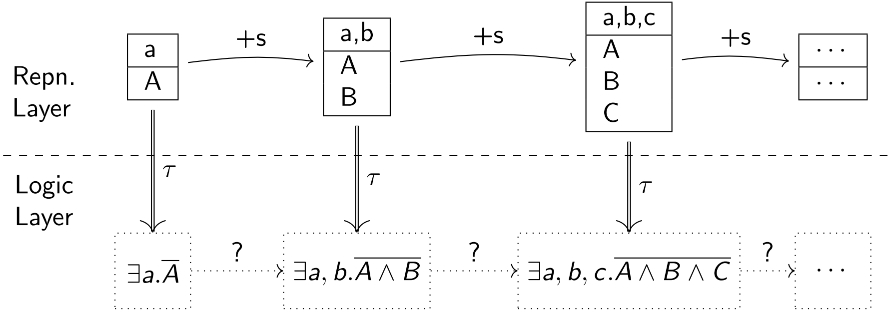
- **Example 1.16 (Existential Quantification is Dynamic).**
  There is a book[1] that Peter does not own. *It$_1$ is a novel.*

$$\begin{array}{|l|}\hline V \\ \hline \mathrm{book}(V) \\ (\neg\mathrm{own}(\mathrm{peter}, V)) \\ \hline\end{array} \otimes \begin{array}{|l|}\hline \\ \hline \mathrm{novel}(V) \\ \hline\end{array} \qquad \exists U.\mathrm{book}(U) \wedge \neg\mathrm{own}(\mathrm{peter}, U) \wedge \mathrm{novel}(U)$$

SOME RIGHTS RESERVED

DRT as a Representational Level

- DRT adds a level to the knowledge representation which provides anchors (the discourse referents) for anaphora and the like.
- Propositional semantics by translation into PL[1]. ("+s" adds a sentence)



- Anaphor resolution works incrementally on the representational level.

SOME RIGHTS RESERVED

A Direct Semantics for DRT (Dyn. Interpretation $\mathcal{I}_\varphi^\delta$)

- **Definition 1.17.** Let $\mathcal{M} = \langle \mathcal{D}, \mathcal{I} \rangle$ be a first-order model, then a state is an assignment from discourse referents into $\mathcal{D}$.
- **Definition 1.18.** Let $\varphi, \psi \colon \mathcal{DR} \to \mathcal{U}$ be states, then we say that ψ extends φ on $\mathcal{X} \subseteq \mathcal{DR}$ (write $\varphi[\mathcal{X}]\psi$), if $\varphi(U) = \psi(U)$ for all $U \notin \mathcal{X}$.
- **Idea:** Conditions as truth values; DRSes as pairs $(\mathcal{X}, \mathcal{S})$ ($\mathcal{S}$ set of states)
- **Definition 1.19 (Meaning of complex formulae).** The value function $\mathcal{I}_\varphi$ for DRT is defined with the help of a dynamic value function $\mathcal{I}_\varphi^\delta$ on DRSs: For conditions:
  - $\mathcal{I}_\varphi(\neg\mathcal{D}) = \mathsf{T}$, if $\mathcal{I}_\varphi^\delta(\mathcal{D})^2 = \emptyset$.
  - $\mathcal{I}_\varphi(\mathcal{D} \vee\!\!\!\vee \mathcal{E}) = \mathsf{T}$, if $\mathcal{I}_\varphi^\delta(\mathcal{D})^2 \neq \emptyset$ or $\mathcal{I}_\varphi^\delta(\mathcal{E})^2 \neq \emptyset$.
  - $\mathcal{I}_\varphi(\mathcal{D} \Rrightarrow \mathcal{E}) = \mathsf{T}$, if for all $\psi \in \mathcal{I}_\varphi^\delta(\mathcal{D})^2$ there is a $\tau \in \mathcal{I}_\varphi^\delta(\mathcal{E})^2$ with $\psi[\mathcal{I}_\varphi^\delta(\mathcal{E})^1]\tau$.

  For DRSs $\mathcal{D}$ we set $\mathcal{I}_\varphi(\mathcal{D}) = \mathsf{T}$, iff $\mathcal{I}_\varphi^\delta(\mathcal{D})^2 \neq \emptyset$, and define
  - $\mathcal{I}_\varphi^\delta(\delta\mathcal{X}.\mathrm{C}) = (\mathcal{X}, \{\psi \mid \varphi[\mathcal{X}]\psi \text{ and } \mathcal{I}_\psi(\mathrm{C}) = \mathsf{T}\})$.
  - $\mathcal{I}_\varphi^\delta(\mathcal{D} \otimes \mathcal{E}) = \mathcal{I}_\varphi^\delta(\mathcal{D} \,;;\, \mathcal{E}) = (\mathcal{I}_\varphi^\delta(\mathcal{D})^1 \cup \mathcal{I}_\varphi^\delta(\mathcal{E})^1, \mathcal{I}_\varphi^\delta(\mathcal{D})^2 \cap \mathcal{I}_\varphi^\delta(\mathcal{E})^2)$

SOME RIGHTS RESERVED

Examples (Computing Direct Semantics)

- **Example 1.20.** *Peter owns a car*

$$\begin{aligned} & \mathcal{I}_\varphi^\delta(\delta U.\mathrm{acar}(U) \wedge \mathrm{own}(\mathrm{peter}, U)) \\ = \;& (\{U\},\{\psi|\varphi[U]\psi \text{ and } \mathcal{I}_\psi(\mathrm{acar}(U) \wedge \mathrm{own}(\mathrm{peter}, U)) = \mathsf{T}\}) \\ = \;& (\{U\},\{\psi|\varphi[U]\psi \text{ and } \mathcal{I}_\psi(\mathrm{acar}(U)) = \mathsf{T} \text{ and } \mathcal{I}_\psi(\mathrm{own}(\mathrm{peter}, U)) = \mathsf{T}\}) \\ = \;& (\{U\},\{\psi|\varphi[U]\psi \text{ and } \psi(U)\in\mathcal{I}(\mathrm{acar}) \text{ and } (\psi(U),\mathrm{peter})\in\mathcal{I}(\mathrm{own})\}) \end{aligned}$$

The set of states $[a/U]$, such that a is a car and is owned by Peter

- **Example 1.21.** For *Peter owns no car* we look at the condition:

$$\begin{aligned} & \mathcal{I}_\varphi(\neg(\delta U.\mathrm{acar}(U) \wedge \mathrm{own}(\mathrm{peter}, U))) = \mathsf{T} \\ \Leftrightarrow \;& \mathcal{I}_\varphi^\delta(\delta U.\mathrm{acar}(U) \wedge \mathrm{own}(\mathrm{peter}, U))^2 = \emptyset \\ \Leftrightarrow \;& (\{U\},\{\psi|\varphi[\mathcal{X}]\psi \text{ and } \psi(U)\in\mathcal{I}(\mathrm{acar}) \text{ and } (\psi(U),\mathrm{peter})\in\mathcal{I}(\mathrm{own})\})^2 = \emptyset \\ \Leftrightarrow \;& \{\psi|\varphi[\mathcal{X}]\psi \text{ and } \psi(U)\in\mathcal{I}(\mathrm{acar}) \text{ and } (\psi(U),\mathrm{peter})\in\mathcal{I}(\mathrm{own})\} = \emptyset \end{aligned}$$

i.e. iff there are no a, that are cars and that are owned by Peter.

SOME RIGHTS RESERVED

12.2 Dynamic Model Generation

Deduction in Dynamic Logics

- Mechanize the dynamic entailment relation (with anaphora)
- Use dynamic deduction theorem to reduce (dynamic) entailment to (dynamic) satisfiability
- Direct Deduction on DRT (or DPL) [Sau93; RG94; MR98]
  - (++) Specialized Calculi for dynamic representations
  - (− −) Needs lots of development until we have efficient implementations
- Translation approach (used in our experiment)
  - (−) Translate to FOL
  - (++) Use off-the-shelf theorem prover (in this case MathWeb)

SOME RIGHTS RESERVED

An Opportunity for Off-The-Shelf ATP?

- **Pro:** ATP is mature enough to tackle applications
  - Current ATP are highly efficient reasoning tools.
  - Full automation is needed for NLP. (ATP as an oracle)
  - ATP as logic engines is one of the initial promises of the field.
- **contra:** ATP are general logic systems
  1. NLP uses other representation formalisms (DRT, Feature Logic,...)
  2. ATP optimized for mathematical (combinatorially complex) proofs.
  3. ATP (often) do not terminate.
- **Experiment:** Use translation approach for 1. to test 2. and 3. [Bla+01] (Wow, it works!)

SOME RIGHTS RESERVED

Excursion: Incrementality in Dynamic Calculi

- For applications, we need to be able to check for
  - satisfiability ($\exists\mathcal{M}.\mathcal{M}\models A$), validity ($\forall\mathcal{M}.\mathcal{M}\models A$) and
  - entailment ($\mathcal{H}\models A$, iff $\mathcal{M}\models\mathcal{H}$ implies $\mathcal{M}\models A$ for all $\mathcal{M}$)
- **Theorem 2.1 (Entailment Theorem).** *$\mathcal{H}, A\models B$, iff $\mathcal{H}\models A\Rightarrow B$.* *(e.g. for first-order logic and DPL)*
- **Theorem 2.2 (Deduction Theorem).** *For most calculi $\mathcal{C}$ we have $\mathcal{H}, A\vdash_{\mathcal{C}}B$, iff $\mathcal{H}\vdash_{\mathcal{C}}A\Rightarrow B$.* *(e.g. for $\mathcal{ND}^1$)*
- **Problem:** Analogue $H_1\otimes\cdots\otimes H_n\models A$ is not equivalent to $\models(H_1\otimes\cdots\otimes H_n)\Rrightarrow A$ in DRT, as $\otimes$ symmetric.
- **Thus** the validity check cannot be used for entailment in DRT.
- **Solution:** Use sequential merge ;; (from DPL) for sentence composition.

SOME RIGHTS RESERVED

Model Generation for Dynamic Logics

- **Problem:** Translation approach is not incremental!
  - For each check, the DRS for the whole discourse has to be translated.
  - Can become infeasible, once discourses get large (e.g. novel).
  - This applies for all other approaches for dynamic deduction too.
- **Idea:** Extend model generation techniques instead!
  - **Remember**: A DRS $\mathcal{D}$ is valid in $\mathcal{M} = \langle \mathcal{D}, \mathcal{I} \rangle$, iff $\mathcal{I}_{\emptyset}^{\delta}(\mathcal{D})^2 \neq \emptyset$.
  - Find a model $\mathcal{M}$ and state φ, such that $\varphi \in \mathcal{I}_{\emptyset}^{\delta}(\mathcal{D})^2$.
  - Adapt first-order model generation technology for that.

SOME RIGHTS RESERVED

Dynamic Herbrand Interpretation

- **Definition 2.3.** We call a model $\mathcal{M} = \langle \mathcal{U}, \mathcal{I}, \mathcal{I}^{\delta}_{\cdot} \rangle$ a dynamic Herbrand interpretation, if $\langle \mathcal{U}, \mathcal{I} \rangle$ is a Herbrand model.
- Can represent $\mathcal{M}$ as a triple $\langle \mathcal{X}, \mathcal{S}, \mathcal{B} \rangle$, where $\mathcal{B}$ is the Herbrand base for $\langle \mathcal{U}, \mathcal{I} \rangle$.
- **Definition 2.4.** $\mathcal{M}$ is called finite, iff $\mathcal{U}$ is finite.
- **Definition 2.5.** $\mathcal{M}$ is minimal, iff for all $\mathcal{M}'$ the following holds: $(\mathcal{B}(\mathcal{M})' \subseteq \mathcal{B}(\mathcal{M})) \Rightarrow \mathcal{M} = \mathcal{M}'$.
- **Definition 2.6.** $\mathcal{M}$ is domain minimal if for all $\mathcal{M}'$ the following holds:

$$\#(\mathcal{U}(\mathcal{M})) \leq \#(\mathcal{U}(\mathcal{M})')$$

SOME RIGHTS RESERVED

Dynamic Model Generation Calculus

- **Definition 2.7.** We use a tableau framework, extend by state information, and rules for DRSes.
  - $$\frac{(\delta U_{\mathbb{A}}.\mathrm{A})^{\mathsf{T}} \quad \mathcal{H}=\{a_1,\ldots,a_n\} \quad w\notin\mathcal{H} \text{ new}}{\begin{array}{c}[a_1/U]\\ ([a_1/U](\mathrm{A}))^{\mathsf{T}}\end{array} \Bigg| \ldots \Bigg| \begin{array}{c}[a_n/U]\\ ([a_n/U](\mathrm{A}))^{\mathsf{T}}\end{array} \Bigg| \begin{array}{c}[w/U]\\ ([w/U](\mathrm{A}))^{\mathsf{T}}\end{array}} RM\,\delta$$
  - Mechanize ;; by adding representation of the second DRS at all leaves. ($\leadsto$ tableau machine)
  - Treat conditions by DRT translation

$$\frac{\neg\mathcal{D}}{\overline{\neg\mathcal{D}}} \qquad \frac{\mathcal{D}\Rrightarrow\mathcal{D}'}{\overline{\mathcal{D}\Rrightarrow\mathcal{D}'}} \qquad \frac{\mathcal{D}\vee\!\!\!\vee\mathcal{D}'}{\overline{\mathcal{D}\vee\!\!\!\vee\mathcal{D}'}}$$

SOME RIGHTS RESERVED

Example: *Peter is a man. No man walks*

- **Example 2.8 (Model Generation).** *Peter is a man. No man walks*

$$\boxed{\mathrm{man}(\mathrm{peter})}$$
$$\boxed{\neg(\delta U.\mathrm{man}(U) \wedge \mathrm{walks}(U))}$$
$$(\mathrm{man}(U) \wedge \mathrm{walks}(U))^{\mathsf{T}}$$
$$(\forall X.\mathrm{man}(X) \wedge \mathrm{walks}(X))^{\mathsf{F}}$$
$$(\mathrm{man}(\mathrm{peter}) \wedge \mathrm{walks}(\mathrm{peter}))^{\mathsf{F}}$$
$$\begin{array}{c|c} \mathrm{man}(\mathrm{peter})^{\mathsf{F}} & \mathrm{walks}(\mathrm{peter})^{\mathsf{F}} \\ \bot & \end{array}$$

Dynamic Herbrand interpretation: $\langle \emptyset, \emptyset, \{\mathrm{man}(\mathrm{peter})^{\mathsf{T}}, \mathrm{walks}(\mathrm{peter})^{\mathsf{F}}\}\rangle$

SOME RIGHTS RESERVED

- **Example 2.9 (Anaphor Resolution).** *A man sleeps. He snores*

$$\boxed{\delta U_{\mathbb{M}\mathrm{an}}.\mathrm{man}(U) \wedge \mathrm{sleeps}(U)}$$
$$[c^1_{\mathbb{M}\mathrm{an}}/U_{\mathbb{M}\mathrm{an}}]$$
$$\mathrm{man}(c^1_{\mathbb{M}\mathrm{an}})^{\mathsf{T}}$$
$$\mathrm{sleeps}(c^1_{\mathbb{M}\mathrm{an}})^{\mathsf{T}}$$
$$\boxed{\delta V_{\mathbb{M}\mathrm{an}}.\mathrm{snores}(V)}$$

$[c^1_{\mathbb{M}\mathrm{an}}/V_{\mathbb{M}\mathrm{an}}]$	$[c^2_{\mathbb{M}\mathrm{an}}/V_{\mathbb{M}\mathrm{an}}]$
$\mathrm{snores}(c^1_{\mathbb{M}\mathrm{an}})^{\mathsf{T}}$	$\mathrm{snores}(c^2_{\mathbb{M}\mathrm{an}})^{\mathsf{T}}$
minimal	deictic

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG
SOME RIGHTS RESERVED

Anaphora with World Knowledge

- **Example 2.10 (Anaphora with World Knowledge).**
  - *Mary is married to Jeff. Her husband is not in town.*
  - $\delta U_{\mathbb{F}}, V_{\mathbb{M}}.U = \mathrm{mary} \wedge \mathrm{married}(U, V) \wedge V = \mathrm{jeff} \mathbin{;;} \delta W_{\mathbb{M}}, W'_{\mathbb{F}}.\mathrm{husband}(W, W') \wedge \neg \mathrm{intown}(W)$
  - World knowledge
    - if a female X is married to a male Y, then Y is X's only husband
    - $\forall X_{\mathbb{F}}, Y_{\mathbb{M}}.\mathrm{married}(X, Y) \Rightarrow \mathrm{husband}(Y, X) \wedge (\forall Z.\mathrm{husband}(Z, X) \Rightarrow Z = Y)$
  - Model generation yields tableau, all branches contain

$$\langle \{U, V, W, W'\}, \{[\mathrm{mary}/U], [\mathrm{jeff}/V], [\mathrm{jeff}/W], [\mathrm{mary}/W']\}, \mathcal{H} \rangle$$

  with

$$\mathcal{H} = \{\mathrm{married}(\mathrm{mary}, \mathrm{jeff})^{\mathsf{T}}, \mathrm{husband}(\mathrm{jeff}, \mathrm{mary})^{\mathsf{T}}, \neg \mathrm{intown}(\mathrm{jeff})^{\mathsf{T}}\}$$

  - they only differ in additional negative facts, e.g. $\mathrm{married}(\mathrm{mary}, \mathrm{mary})^{\mathsf{F}}$.

SOME RIGHTS RESERVED

Model Generation models Discourse Understanding

- Conforms with psycholinguistic findings:
  - Zwaan& Radvansky [ZR98]: listeners not only represent logical form, but also models containing referents.
  - deVega [de 95]: online, incremental process.
  - Singer [Sin94]: enriched by background knowledge.
  - Glenberg et al. [GML87]: major function is to provide basis for anaphor resolution.

SOME RIGHTS RESERVED

Chapter 13
Propositional Attitudes and Modalities

SOME RIGHTS RESERVED

13.1 Introduction

SOME RIGHTS RESERVED

Modalities and Propositional Attitudes

- **Definition 1.1.** Modality is a feature of language that allows for communicating things about, or based on, situations which need not be actual.
- **Definition 1.2.** Modality is signaled by grammatical expressions (called moods) that express a speaker's general intentions and commitment to how believable, obligatory, desirable, or actual an expressed proposition is.
- **Example 1.3.** Data on modalities (moods in red)
  - A *probably holds*, (possibilistic)
  - *it has always been the case that* A, (temporal)
  - *it is well-known that* A, (epistemic)
  - A *is allowed/prohibited*, (deontic)
  - A *is provable*, (provability)
  - A *holds after the program P terminates*, (program)
  - A *hods during the execution of P*. (dito)
  - *it is necessary that* A, (aletic)
  - *it is possible that* A, (dito)

SOME RIGHTS RESERVED

Modeling Modalities and Propositional Attitudes

- **Example 1.4.** Again, the pattern from above:
  - *it is necessary that Peter knows logic* (A = Peter knows logic)
  - *it is possible that John loves logic*, (A = John loves logic)
- **Observation:** All of the red parts above modify the clause/sentence A. We call them modalities.
- **Definition 1.5 (A related Concept from Philosophy).** A propositional attitude is a mental state held by an agent toward a proposition.
- **Question:** But how to model this in logic?
- **Idea:** New sentence-to-sentence operators for *necessary* and *possible*. (extend existing logics with them.)
- **Observation:** A *is necessary*, iff $\neg$A *is impossible*.
- **Definition 1.6.** A modal logic is a logical system that has logical constants that model modalities.

SOME RIGHTS RESERVED

History of Modal Logic

- Aristoteles studies the logic of necessity and possibility
- Diodorus: temporal modalities
  - possible: *is true or will be*
  - necessary: *is true and will never be false*
- Clarence Irving Lewis 1918 [Lew18] (Systems $S1, \ldots, S5$)
  - strict implication $I(\mathrm{A} \wedge \mathrm{B})$ (I for "impossible")
- Kurt Gödel 1932: Modal logic of provability (S4) [Göd32]
- Saul Kripke 1959-63: Possible worlds semantics [Kri63]
- Vaugham Pratt 1976: Dynamic Program Logic [Pra76]
- $\vdots$

SOME RIGHTS RESERVED

Basic Modal Logics ($\mathcal{ML}^0$ and $\mathcal{ML}^1$)

- **Definition 1.7.** Propositional modal logic $\mathcal{ML}^0$ extends propositional logic with two new logical constants: $\Box$ for necessity and $\Diamond$ for possibility.($\Diamond A = \neg(\Box\neg A)$)
- **Observation:** Nothing hinges on the fact that we use propositional logic!
- **Definition 1.8.** First-order modal logic $\mathcal{ML}^1$ extends first-order logic with two new logical constants: $\Box$ for necessity and $\Diamond$ for possibility.
- **Example 1.9.** We interpret
  1. *Necessarily, every mortal will die.* as $\Box(\forall X.\mathrm{mortal}(X) \Rightarrow \mathrm{willdie}(X))$
  2. *Possibly, something is immortal.* as $\Diamond(\exists X.\neg\mathrm{mortal}(X))$
- **Questions:** What do $\Box$ and $\Diamond$ mean? How do they behave?

SOME RIGHTS RESERVED

Epistemic and Doxastic Modality

- **Definition 1.10.** Modal sentences can convey information about the speaker's state of knowledge (epistemic state) or belief (doxastic state).
- **Example 1.11.** We might paraphrase sentence (epposs) as (3):
  1. A: *Where's John?*
  2. B: *He might be in the library.*
  3. B': *It is consistent with the speaker's knowledge that John is in the library.*
- **Definition 1.12.** We way that a world w is an epistemic possibility for an agent B if it could be consistent with B's knowledge.
- **Definition 1.13.** An epistemic logic is one that models the epistemic state of a speaker. Doxastic logic does the same for the doxastic state.
- **Definition 1.14.** In deontic modal logic, we interpret the accessibility relation $\mathcal{R}$ as epistemic accessibility:
  - With this $\mathcal{R}$, represent B's utterance as $\Diamond\mathrm{inlib}(j)$.
  - Similarly, represent *John must be in the library.* as $\Box\mathrm{inlib}(j)$.
- **Question:** If $\mathcal{R}$ is epistemic accessibility, what properties should it have?

SOME RIGHTS RESERVED

Deontic modality

- **Definition 1.15.** Deontic modality is a modality that indicates how the world ought to be according to certain norms, expectations, speaker desire, etc.
- **Definition 1.16.** Deontic modality has the following subcategories
  - Commissive modality (the speaker's commitment to do something, like a promise or threat): e.g. *I shall help you*.
  - Directive modality (commands, requests, etc.): e.g. *Come!*, *Let's go!*, *You've got to taste this curry!*
  - Volitive modality (wishes, desires, etc.): *If only I were rich!*
- **Question:** If we want to interpret $\Box \mathrm{runs}(j)$ as *It is required that John runs* (or, more idiomatically, as *John must run*), what formulae should be valid on this interpretation of the operators? (This is for homework!)

SOME RIGHTS RESERVED

13.2 Semantics for Modal Logics

SOME RIGHTS RESERVED

Semantics of ML^0

- **Definition 2.1.** We use a set $\mathcal{W}$ of possible worlds, and a accessibility relation $\mathcal{R} \subseteq \mathcal{W} \times \mathcal{W}$: if $\mathcal{R}(v, w)$, then we say that w is accessible from v.
- **Example 2.2.** $\mathcal{W} = \mathbb{N}$ with $\mathcal{R} = \{\langle n, n+1\rangle | n \in \mathbb{N}\}$. (temporal logic)
- **Definition 2.3.** Variable assignment $\varphi\colon \mathcal{V}_0 \times \mathcal{W} \to \mathcal{D}_0$ assigns values to variables in a given possible world.
- **Definition 2.4.** Value function $\mathcal{I}_{\cdot}^{\cdot}\colon \mathcal{W} \times wff_0() \to \mathcal{D}_0$ (assigns values to formulae in a possible world)
  - $\mathcal{I}_\varphi^w(V) = \varphi(w, V)$
  - $\mathcal{I}_\varphi^w(\neg \mathrm{A}) = \mathsf{T}$, iff $\mathcal{I}_\varphi^w(\mathrm{A}) = \mathsf{F}$ ($\wedge$ analogous)
  - $\mathcal{I}_\varphi^w(\Box \mathrm{A}) = \mathsf{T}$, iff $\mathcal{I}_\varphi^{w'}(\mathrm{A}) = \mathsf{T}$ for all $w' \in \mathcal{W}$ with $w \mathcal{R} w'$.
- **Definition 2.5.** We call a triple $\mathcal{M} := \langle \mathcal{W}, \mathcal{R}, \mathcal{I}\rangle$ a Kripke model.

SOME RIGHTS RESERVED

Accessibility Relations. E.g. for Temporal Modalities

- **Example 2.6 (Temporal Worlds with Ordering).** Let $\langle\mathcal{W},\circ,<,\subseteq\rangle$ an interval time structure, then we can use $\langle\mathcal{W},<\rangle$ as a Kripke models. Then PAST becomes a modal operator.
- **Example 2.7.** Suppose we have $i<j$ and $j<k$. Then intuitively, if *Jane is laughing* is true at i, then *Jane laughed* should be true at j and at k, i.e. $\mathcal{I}_\varphi^w(j)\mathrm{PAST}(\mathrm{laughs}(j))$ and $\mathcal{I}_\varphi^w(k)\mathrm{PAST}(\mathrm{laughs}(j))$.
  But this holds only if "$<$" is transitive. (which it is!)
- **Example 2.8.** Here is a clearly counter-intuitive claim: For any time i and any sentence A, if $\mathcal{I}_\varphi^w(i)\mathrm{PRES}(\mathrm{A})$ then $\mathcal{I}_\varphi^w(i)\mathrm{PAST}(\mathrm{A})$.
  (For example, the truth of *Jane is at the finish line* at i implies the truth of *Jane was at the finish line* at i.)
  But we would get this result if we allowed $<$ to be reflexive. ($<$ is irreflexive)
- Treating tense modally, we obtain reasonable truth conditions.

SOME RIGHTS RESERVED

Modal Axioms (Propositional Logic)

- **Definition 2.9.** Necessitation: $\dfrac{A}{\Box A}N$
- **Definition 2.10 (Normal Modal Logics).**

System	Axioms	Accessibility Relation
$\mathbb{K}$	$\Box(A \Rightarrow B) \Rightarrow (\Box A \Rightarrow \Box B)$	general
$\mathbb{T}$	$\mathbb{K} + \Box A \Rightarrow A$	reflexive
$\mathbb{S}4$	$\mathbb{T} + \Box A \Rightarrow \Box\Box A$	reflexive + transitive
$\mathbb{B}$	$\mathbb{T} + \Diamond\Box A \Rightarrow A$	reflexive + symmetric
$\mathbb{S}5$	$\mathbb{S}4 + \Diamond A \Rightarrow \Box\Diamond A$	equivalence relation

SOME RIGHTS RESERVED

$\mathbb{K}$ Theorems

- **Observation 2.11.** $\Box(A \wedge B) \models \Box A \wedge \Box B$ *in* $\mathbb{K}$.
- **Observation 2.12.** $A \Rightarrow B \models \Box A \Rightarrow \Box B$ *in* $\mathbb{K}$.
- **Observation 2.13.** $A \Rightarrow B \models \Diamond A \Rightarrow \Diamond B$ *in* $\mathbb{K}$.

SOME RIGHTS RESERVED

Translation to First-Order Logic

- **Question:** Is modal logic more expressive than predicate logic?
- **Answer:** Very rarely! (usually can be translated)
- **Definition 2.14.** Translation τ from ML into PL^1, (so that the diagram commutes)

$$\begin{array}{ccc} \text{Kripke-Sem.} & \xrightarrow{\overline{\tau}} & \text{Tarski-Sem.} \\ \mathcal{I}_\varphi^w \uparrow & & \uparrow \mathcal{I}_\varphi \\ \text{modal logic} & \xrightarrow{\tau} & \text{predicate logic} \end{array}$$

- **Idea:** Axiomatize Kripke models in PL^1. (diagram is simple consequence)
- **Definition 2.15.** A logic morphism $\Theta\colon \mathcal{L} \to \mathcal{L}'$ is called
  - correct, iff $\exists \mathcal{M}.\mathcal{M} \models \Phi$ implies $\exists \mathcal{M}'.\mathcal{M}' \models' \Theta(\Phi)$.
  - complete, iff $\exists \mathcal{M}'.\mathcal{M}' \models' \Theta(\Phi)$ implies $\exists \mathcal{M}.\mathcal{M} \models \Phi$.

SOME RIGHTS RESERVED

Modal Logic Translation (formal)

- **Definition 2.16.** The standard translation τ_w from modal logics to first-order logic is given by the following process:
  - Extend all function constants by a "world argument": $\overline{f}\in\Sigma^f_{k+1}$ for every $f\in\Sigma^f_k$
  - for predicate constants accordingly.
  - insert the "translation world" there: e.g. $\tau_w(f(a,b)) = \overline{f}(w,\overline{a}(w),\overline{b}(w))$.
  - New predicate constant $\mathcal{R}$ for the accessibility relation.
  - New constant s for the "start world".
  - $\tau_w(\Box A) = \forall w'.w\mathcal{R}w' \Rightarrow \tau_{w'}(A)$.
  - Use all axioms from the respective correspondence theory.
- **Definition 2.17 (Alternative).** Functional translations, if $\mathcal{R}$ associative:
  - New function constant $f_{\mathcal{R}}$ for the accessibility relation.
  - Revise the standard translation by one of the following
    - $\tau_w(\Box A) = \forall w'.w = f_{\mathcal{R}}(w') \Rightarrow \tau_w(A)$. (naive solution)
    - $\tau_{f_{\mathcal{R}}(w)}(\Box A) = \tau_w(A)$ (better for mechanizing [Ohl88])

SOME RIGHTS RESERVED

Translation (continued)

- **Theorem 2.18.** *$\tau_s\colon \mathcal{ML}^0 \to \mathcal{PL}^0$ is correct and complete.*
- *Proof:* show that $\exists\mathcal{M}.\mathcal{M} \models \Phi$ iff $\exists\mathcal{M}'.\mathcal{M}' \models \tau_s(\Phi)$
  1. Let $\mathcal{M} = \langle\mathcal{W}, \mathcal{R}, \varphi\rangle$ with $\mathcal{M} \models \mathrm{A}$
  2. chose $\mathcal{M} = \langle\mathcal{W}, \mathcal{I}'\rangle$, such that $\mathcal{I}(\overline{p}) = \varphi(p)\colon \mathcal{W} \to \{\mathsf{T}, \mathsf{F}\}$ and $\mathcal{I}(r) = \mathcal{R}$.

  we prove $\mathcal{M} \models_\psi \tau_w(\mathrm{A})'$ for $\psi = Id_\mathcal{W}$ by structural induction over A.
  3. $\mathrm{A} = P$
     1. $\mathcal{I}_\psi(\tau_w(\mathrm{A})) = \mathcal{I}_\psi(\overline{p}(w)) = \mathcal{I}(\overline{p}(w)) = \varphi(P, w) = \mathsf{T}$
  4. $\mathrm{A} = \neg\mathrm{B}$, $\mathrm{A} = \mathrm{B} \wedge \mathrm{C}$ *trivial by IH.*
  5. $\mathrm{A} = \Box\mathrm{B}$
     1. $\mathcal{I}_\psi(\tau_w(\mathrm{A})) = \mathcal{I}_\psi(\forall w.r(w, v) \Rightarrow \tau_v(\mathrm{B})) = \mathsf{T}$, if $\mathcal{I}_\psi(r(w, v)) = \mathsf{F}$ or $\mathcal{I}_\psi(\tau_v(\mathrm{B})) = \mathsf{T}$ for all $v \in \mathcal{W}$.
     2. $\mathcal{M} \models_\psi \tau_{v'}(\mathrm{B})$ so by IH $\mathcal{M} \models^v \mathrm{B}$.
     3. so $\mathcal{M} \models_\psi \tau_w(\mathrm{A})'$.

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG
SOME RIGHTS RESERVED

Modal Logic (References)

- G. E. Hughes und M. M. Cresswell: *A companion to Modal Logic*, University Paperbacks, Methuen (1984) [HC84].
- David Harel: *Dynamic Logic*, Handbook of Philosophical Logic, D. Gabbay, Hrsg. Reidel (1984) [Har84].
- Johan van Benthem: *Language in Action, Categories, Lambdas and Dynamic Logic*, North Holland (1991) [Ben91].
- Reinhard Muskens, Johan van Benthem, Albert Visser, *Dynamics*, in *Handbook of Logic and Language*, Elsevier, (1995) [MBV95].
- Blackburn, DeRijke, Vedema: *Modal Logic*; 2001 [BRV01]. look at the chapter "Guide to the literature" in the end.

SOME RIGHTS RESERVED

13.3 A Multiplicity of Modalities $\rightsquigarrow$ Multimodal Logic

SOME RIGHTS RESERVED

A Multiplicity of Modalities

- Epistemic (knowledge and belief) modalities must be relativized to an individual
  - *Peter knows that Trump is lying habitually.*
  - *John believes that Peter knows that Trump is lying habitually.*
  - *You must take the written drivers' exam to be admitted to the practical test.*
- Similarly, we find in natural language expressions of necessity and possibility relative to many different kinds of things.
- Consider the deontic (obligatory/permissible) modalities
  - *[Given the university's rules] Jane can take that class.*
  - *[Given her intellectual ability] Jane can take that class.*
  - *[Given her schedule] Jane can take that class.*
  - *[Given my desires] I must meet Henry.*
  - *[Given the requirements of our plan] I must meet Henry.*
  - *[Given the way things are] I must meet Henry [every day and not know it].*
- Many different sorts of modality, sentences are multiply ambiguous towards which one.

SOME RIGHTS RESERVED

Multimodal Logics

- **Definition 3.1.** A multi modal logic provides operators for multiple modalities: $[1], [2], [3], \ldots, \langle 1\rangle, \langle 2\rangle, \ldots$
- **Definition 3.2.** Multi modal Kripke models provide multiple accessibility relations $\mathcal{R}_1, \mathcal{R}_2, \ldots \subseteq \mathcal{W} \times \mathcal{W}$.
- **Definition 3.3.** The value function in logic generalizes the clause for $\Box$ in ML^0 to
  - $\mathcal{I}_\varphi^w([i]\mathrm{A}) = \mathsf{T}$, iff $\mathcal{I}_\varphi^{w'}(\mathrm{A}) = \mathsf{T}$ for all $w' \in \mathcal{W}$ with $w\mathcal{R}_i w'$.
- **Example 3.4 (Epistemic Logic: talking about knowing/believing).**
  $[peter]\langle klaus\rangle\mathrm{A}$ (Peter knows that Klaus considers A possible)
- **Example 3.5 (Program Logic: talking about programs).**
  $[X{:=}\mathrm{A}][Y{:=}\mathrm{A}]X = Y$ (after assignments, the values of X and Y are equal)

SOME RIGHTS RESERVED

13.4 Dynamic Logic for Imperative Programs

SOME RIGHTS RESERVED

Dynamic Program Logic (DL)

- Modal logics for argumentation about imperative, non-deterministic programs.
- **Idea:** Formalize the traditional argumentation about program correctness: tracing the variable assignments (state) across program statements.
- **Example 4.1 (Fibonacci).**
  Consider the following (imperative) program that computes $\mathrm{Fib}(X)$ as the value of Z:
  $\alpha{:=}\langle Y, Z\rangle{:=}\langle 1, 1\rangle$; while $X \neq 0$ do $\langle X, Y, Z\rangle{:=}\langle X-1, Z, Y+Z\rangle$ end
  - **States** for the "input" $X = 4$: $\langle 4, \_, \_\rangle, \langle 4, 1, 1\rangle, \langle 3, 1, 2\rangle, \langle 2, 2, 3\rangle, \langle 1, 3, 5\rangle, \langle 0, 5, 8\rangle$
  - **Correctness?** For positive X, running α with input $\langle X, \_, \_\rangle$ we end with $\langle 0, F_{(X-1)}, F_X\rangle$
  - **Termination?** α does not terminate on input $\langle -1, \_, \_\rangle$.

SOME RIGHTS RESERVED

Multi-Modal Logic fits well

- **Observation:** Multi modal logic fits well
  - States as possible worlds, program statements as accessibility relations.
  - Two syntactic categories: programs α and formulae A.
  - Interpret $[\alpha]$A as *If α terminates, then A holds afterwards*
  - Interpret $\langle\alpha\rangle$A as α *terminates and A holds afterwards*.
- **Example 4.2.** Assertions about Fibonacci number (α)
  - $\forall X, Y.[\alpha]Z = \mathrm{Fib}(X)$
  - $\forall X, Y.(X \geq 0) \Rightarrow \langle\alpha\rangle Z = \mathrm{Fib}(X)$

SOME RIGHTS RESERVED

Levels of Description in Dynamic Logic

- Propositional dynamic logic (DL^0) (independent of variable assignments)
  - $\models ([\alpha]A \wedge [\alpha]B) \Leftrightarrow ([\alpha](A \wedge B))$
  - $\models ([\text{while } A \vee B \text{ do } \alpha \text{ end}]C) \Leftrightarrow ([\text{while } A \text{ do } \alpha \text{ end} ; \text{while } B \text{ do } \alpha ; \text{while } A \text{ do } \alpha \text{ end end}]C)$
- First-order program logic (DL^1) (function, predicates uninterpreted)
  - $\models p(f(X)) \Rightarrow g(Y, f(X)) \Rightarrow \langle (Z{:=}f(X)) \rangle p(Z, g(Y, Z))$
  - $\models Z = Y \wedge (\forall X.f(g(X)) = X) \Rightarrow [\text{while } p(Y) \text{ do } Y{:=}g(Y) \text{ end}]\langle \text{while } Y \neq Z \text{ do } Y{:=}f(Y) \text{ end} \rangle T$
- DL^1 with interpreted functions, predicates (maybe some other time)
  - $\forall X.\langle \text{while } X \neq 1 \text{ do if } even(X) \text{ then} X{:=}\frac{X}{2} \text{ else } X{:=}3X+1 \text{ end} \rangle T$

SOME RIGHTS RESERVED

DL^0 Syntax

- **Definition 4.3.** Propositional dynamic logic (DL^0) is PL^0 extended by
  - program variables $\mathcal{V}_\pi = \{\alpha, \beta, \gamma, \ldots\}$,
  - modalities $[\alpha]$, $\langle\alpha\rangle$.
  - program constructors $\Sigma^\pi = \{;, \cup, *, ?\}$ (minimal set)

$\alpha\,;\,\beta$	execute first α, then β	sequence
$\alpha \cup \beta$	execute (non-deterministically) either α or β	distribution
$*\alpha$	(non-deterministically) repeat α finitely often	iteration
$A?$	proceed if $\models A$, else error	test

- **Idea:** Standard program primitives as derived concepts

Construct	as
if A thenα else β	$(A?\,;\,\alpha) \cup (\neg A?\,;\,\beta)$
while A do α end	$*(A?\,;\,\alpha)\,;\,\neg A?$
repeat α until A end	$*(\alpha\,;\,\neg A?)\,;\,A?$

SOME RIGHTS RESERVED

DL^0 Semantics

- **Definition 4.4.** A model for DL^0 consists of a set $\mathcal{W}$ of possible worlds called states for DL^0.
- **Definition 4.5.** DL^0 variable assignments come in two parts:
  - $\varphi\colon \mathcal{V}_0 \times \mathcal{W} \to \mathcal{D}_0$ (for propositional variables)
  - $\pi\colon \mathcal{V}_\pi \to \mathcal{P}(\mathcal{W} \times \mathcal{W})$ (maps program variables to accessibility relations)
- **Definition 4.6.** The meaning of complex formulae is given by the following value function $\mathcal{I}_{\varphi,\pi}^{w}\colon wff_0(\mathcal{V}_0) \to \mathcal{D}_0$:
  - $\mathcal{I}_{\varphi,\pi}^{w}(V) = \varphi(w, V)$ for $V \in \mathcal{V}_0$ and $\mathcal{I}_{\varphi,\pi}^{w}(\alpha) = \pi(\alpha)$ for $\alpha \in \mathcal{V}_\pi$.
  - $\mathcal{I}_{\varphi,\pi}^{w}(\neg A) = \mathsf{T}$ iff $\mathcal{I}_{\varphi,\pi}^{w}(A) = \mathsf{F}$
  - $\mathcal{I}_{\varphi,\pi}^{w}([\alpha]A) = \mathsf{T}$ iff $\mathcal{I}_{\varphi,\pi}^{w'}(A) = \mathsf{T}$ for all $w' \in \mathcal{W}$ with $w\mathcal{I}_{\varphi,\pi}^{w}(\alpha)w'$.
  - $\mathcal{I}_{\varphi,\pi}^{w}(\alpha) = \pi(\alpha)$. (program variable by assignment)
  - $\mathcal{I}_{\varphi,\pi}^{w}(\alpha\,;\beta) = \mathcal{I}_{\varphi,\pi}^{w}(\beta) \circ \mathcal{I}_{\varphi,\pi}^{w}(\alpha)$ (sequence by composition)
  - $\mathcal{I}_{\varphi,\pi}^{w}(\alpha \cup \beta) = \mathcal{I}_{\varphi,\pi}^{w}(\alpha) \cup \mathcal{I}_{\varphi,\pi}^{w}(\beta)$ (distribution by union)
  - $\mathcal{I}_{\varphi,\pi}^{w}(*\alpha) = \mathcal{I}_{\varphi,\pi}^{w}(\alpha)^*$ (iteration by reflexive transitive closure)
  - $\mathcal{I}_{\varphi,\pi}^{w}(A?) = \{\langle w, w\rangle | \mathcal{I}_{\varphi,\pi}^{w}(A) = \mathsf{T}\}$ (test by subset of identity relation)

SOME RIGHTS RESERVED

First-Order Program Logic (DL^1)

- **Observation:** Imperative programs contain variables, constants, functions and predicates (uninterpreted), but no program variables. The main operation is variable assignment.
- **Idea:** Make a multi modal logic in the spirit of DL^0 that features all of these for a deeper understanding.
- **Definition 4.7.** First-order program logic (DL^1) combines the features of PL^1, DL^0 without program variables, with the following two assignment operators:
  - nondeterministic assignment $X{:=}?$
  - deterministic assignment $X{:=}A$
- **Example 4.8.** $\models p(f(X)) \Rightarrow g(Y, f(X)) \Rightarrow \langle Z{:=}f(X)\rangle p(Z, g(Y, Z))$ in DL^1.
- **Example 4.9.** In DL^1 we have
$\models Z = Y \wedge (\forall X.p(f(g(X)) = X)) \Rightarrow [\mathrm{while}\ p(Y)\ \mathrm{do}\ Y{:=}g(Y)\ \mathrm{end}]\langle \mathrm{while}\ Y \neq Z\ \mathrm{do}\ Y{:=}f(Y)\ \mathrm{end}\rangle T$

SOME RIGHTS RESERVED

DL^1 Semantics

- **Definition 4.10.** Let $\mathcal{M} = \langle\mathcal{D},\mathcal{I}\rangle$ be a first-order model then the states (possible worlds) are variable assignments: $\mathcal{W} = \{\varphi|\varphi\colon \mathcal{V}_\iota\to\mathcal{D}\}$
- **Definition 4.11.** For a set $\mathcal{X}$ of variables, write $\varphi[\mathcal{X}]\psi$, iff $\varphi(X) = \psi(X)$ for all $X\notin\mathcal{X}$.
- **Definition 4.12.** The meaning of complex formulae is given by the following value function $\mathcal{I}^w_\varphi\colon \mathit{wff}_o(\Sigma,\mathcal{V}_\iota)\to\mathcal{D}_0$
  - $\mathcal{I}^w_\varphi(\mathrm{A}) = \mathcal{I}_\varphi(\mathrm{A})$ if A term or atom.
  - $\mathcal{I}^w_\varphi(\neg\mathrm{A}) = \mathrm{T}$ iff $\mathcal{I}^w_\varphi(\mathrm{A}) = \mathrm{F}$
  - $\vdots$
  - $\mathcal{I}^w_\varphi(X{:=}?) = \{\langle\varphi,\psi\rangle|\varphi[X]\psi\}$
  - $\mathcal{I}^w_\varphi(X{:=}\mathrm{A}) = \{\langle\varphi,\psi\rangle|\varphi[X]\psi$ and $\psi(X) = \mathcal{I}_\varphi(\mathrm{A})\}$.
- **Observation 4.13 (Substitution and Quantification).** *We have*
  - $\mathcal{I}_\varphi([X{:=}\mathrm{A}]\mathrm{B}) = \mathcal{I}_{(\varphi,[\mathcal{I}_\varphi(\mathrm{A})/X])}(\mathrm{B})$
  - $\forall X.\mathrm{A} = [X{:=}?]\mathrm{A}$.
- Thus substitutions and quantification are definable in DL^1.

SOME RIGHTS RESERVED

Natural Language as Programming Languages

- **Question:** Why is dynamic program logic interesting in a natural language course?
- **Answer:** There are fundamental relations between dynamic (discourse) logics and dynamic program logics.
- **David Israel:** "*Natural languages are programming languages for mind*" [Isr93]

SOME RIGHTS RESERVED

Chapter 14
Some Issues in the Semantics of Tense

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG
SOME RIGHTS RESERVED

Tense as a Deictic Element

- **Goal:** capturing the truth conditions and the logical form of sentences of English.
- **Clearly:** the following three sentences have different truth conditions.
  1. *Jane saw George.*
  2. *Jane sees George.*
  3. *Jane will see George.*
- **Observation 0.1.** *Tense is a deictic element, i.e. its interpretation requires reference to something outside the sentence itself.*
- **Remark:** Often, in particular in the case of monoclausal sentences occurring in isolation, as in our examples, this "something" is the speech time.
- **Idea:** make use of the reference time *now*:
  - *Jane saw George* is true at a time iff *Jane sees George* was true at some point in time before now.
  - *Jane will see George* is true at a time iff *Jane sees George* will be true at some point in time after now.

SOME RIGHTS RESERVED

A Simple Semantics for Tense

- **Problem:** the meaning of *Jane saw George* and *Jane will see George* is defined in terms of *Jane sees George*.
  $\leadsto$ We need the truth conditions of the present tense sentence.
- **Idea:** *Jane sees George* is true at a time iff Jane sees George at that time.
- **Implementation:** Postulate tense operators as sentential operators (expressions of type $\mathrm{prop} \to \mathrm{prop}$). Interpret
  1. *Jane saw George* as $\mathrm{PAST}(\mathrm{see}(g, j))$,
  2. *Jane sees George* as $\mathrm{PRES}(\mathrm{see}(g, j))$, and
  3. *Jane wil see George* as $\mathrm{FUT}(\mathrm{see}(g, j))$.

SOME RIGHTS RESERVED

Models and Evaluation for a Tensed Language

- **Problem:** The interpretations of constants vary over time.
- **Idea:** Introduce times into our models, and let the interpretation function give values of constants at a time. Relativize the valuation function to times
- **Idea:** We will consider temporal structures, where denotations are constant on intervals.
- **Definition 0.2.** Let $I \subseteq \{[i,j] | i,j \in \mathbb{R}\}$ be a set of real intervals, then we call $\langle I, \circ, <, \subseteq \rangle$ an interval time structure, where for intervals $i := [i_l, i_l]$ and $j := [l_l, j_r]$ we say that
  - i and j overlap (written $i \circ j$), iff $l_l \leq ir$,
  - i precedes j (written $i < j$), iff $ir \leq l_l$, and
  - i is contained in j (written $i \subseteq j$), iff $l_l \leq i_l$ and $ir \leq j_r$.
- **Definition 0.3.** A temporal model is a triple $\langle \mathcal{D}, \mathbb{I}, \mathcal{I} \rangle$, where
  - $\mathcal{D}$ is a set called the domain,
  - $\mathbb{I}$is a interval time structure, and
  - $\mathcal{I} : \mathbb{I} \times \Sigma_{\mathcal{T}} \to \mathcal{D}$ an interpretation function.

SOME RIGHTS RESERVED

Interpretation rules for the temporal operators

- **Definition 0.4.** For the value function $\mathcal{I}_i(\varphi)\cdot$ we only redefine the clause for constants:
  - $\mathcal{I}_i(\varphi)c := \mathcal{I}(i, c)$
  - $\mathcal{I}_i(\varphi)X := \varphi(X)$
  - $\mathcal{I}_i(\varphi)\mathrm{FA} := \mathcal{I}_i(\varphi)\mathrm{F}(\mathcal{I}_i(\varphi)\mathrm{A})$.
- **Definition 0.5.** We define the meaning of the tense operators
  1. $\mathcal{I}_i(\varphi)\mathrm{PRES}(\Phi) = \mathrm{T}$, iff $\mathcal{I}_i(\varphi)\Phi = \mathrm{T}$.
  2. $\mathcal{I}_i(\varphi)\mathrm{PAST}(\Phi) = \mathrm{T}$ iff there is an interval $j \in I$ with $j < i$ and $\mathcal{I}_j(\varphi)\Phi = \mathrm{T}$.
  3. $\mathcal{I}_i(\varphi)\mathrm{FUT}(\Phi) = \mathrm{T}$ iff there is an interval $j \in I$ with $i < j$ and $\mathcal{I}_j(\varphi)\Phi = \mathrm{T}$.

SOME RIGHTS RESERVED

Complex tenses in English

- How do we use this machinery to deal with complex tenses in English?
  - Past of past (pluperfect): *Jane had left (by the time I arrived)*.
  - Future perfect: *Jane will have left (by the time I arrive)*.
  - Past progressive: *Jane was going to leave (when I arrived)*.
- Perfective vs. imperfective
  - *Jane left.*
  - *Jane was leaving.*
- How do the truth conditions of these sentences differ? **Standard observation:** Perfective indicates a completed action, imperfective indicates an incomplete or ongoing action. This becomes clearer when we look at the "creation predicates" like *build a house* or *write a book*
  - *Jane built a house.* entails: *There was a house that Jane built.*
  - *Jane was building a house.* does not entail that *there was a house that Jane built.*

SOME RIGHTS RESERVED

Future readings of present tense

- New Data;
  1. *Jane leaves tomorrow.*
  2. *Jane is leaving tomorrow.*
  3. ?? *It rains tomorrow.*
  4. ?? *It is raining tomorrow.*
  5. ?? *The dog barks tomorrow.*
  6. ??*The dog is barking tomorrow.*
- Future readings of present tense appear to arise only when the event described is planned, or planable, either by the subject of the sentence, the speaker, or a third party.

SOME RIGHTS RESERVED

Sequence of Tense

- *George said that Jane was laughing.*
  - Reading 1: George said "*Jane is laughing.*" I.e. saying and laughing co-occur. So past tense in subordinate clause is past of utterance time, but not of main clause reference time.
  - Reading 2: George said "*Jane was laughing.*" I.e. laughing precedes saying. So past tense in subordinate clause is past of utterance time and of main clause reference time.

SOME RIGHTS RESERVED

Sequence of Tense

- *George said that Jane was laughing.*
  - Reading 1: George said "*Jane is laughing.*" I.e. saying and laughing co-occur. So past tense in subordinate clause is past of utterance time, but not of main clause reference time.
  - Reading 2: George said "*Jane was laughing.*" I.e. laughing precedes saying. So past tense in subordinate clause is past of utterance time and of main clause reference time.
- *George saw the woman who was laughing.*
  - How many readings?

SOME RIGHTS RESERVED

Sequence of Tense

- *George said that Jane was laughing.*
  - Reading 1: George said "*Jane is laughing.*" I.e. saying and laughing co-occur. So past tense in subordinate clause is past of utterance time, but not of main clause reference time.
  - Reading 2: George said "*Jane was laughing.*" I.e. laughing precedes saying. So past tense in subordinate clause is past of utterance time and of main clause reference time.
- *George saw the woman who was laughing.*
  - How many readings?
- *George will say that Jane is laughing.*
  - Reading 1: George will say "*Jane is laughing.*" Saying and laughing co-occur, but both saying and laughing are future of utterance time. So present tense in subordinate clause indicates futurity relative to utterance time, but not to main clause reference time.
  - Reading 2: Laughing overlaps utterance time and saying (by George). So present tense in subordinate clause is present relative to utterance time *and* main clause reference time.

SOME RIGHTS RESERVED

Sequence of Tense

- *George will see the woman who is laughing.*
  - How many readings?
- Note that in all of the above cases, the predicate in the subordinate clause describes an event that is extensive in time. Consider readings when subordinate event is punctual.

SOME RIGHTS RESERVED

Sequence of Tense

- *George will see the woman who is laughing.*
  - How many readings?
- Note that in all of the above cases, the predicate in the subordinate clause describes an event that is extensive in time. Consider readings when subordinate event is punctual.
- *George said that Mary fell.*
  - Falling must precede George's saying.

SOME RIGHTS RESERVED

Sequence of Tense

- *George will see the woman who is laughing.*
  - How many readings?
- Note that in all of the above cases, the predicate in the subordinate clause describes an event that is extensive in time. Consider readings when subordinate event is punctual.
- *George said that Mary fell.*
  - Falling must precede George's saying.
- *George saw the woman who fell.*
  - Same three readings as before: falling must be past of utterance time, but could be past, present or future relative to seeing (i.e main clause reference time).

SOME RIGHTS RESERVED

Sequence of Tense

- *George will see the woman who is laughing.*
  - How many readings?
- Note that in all of the above cases, the predicate in the subordinate clause describes an event that is extensive in time. Consider readings when subordinate event is punctual.
- *George said that Mary fell.*
  - Falling must precede George's saying.
- *George saw the woman who fell.*
  - Same three readings as before: falling must be past of utterance time, but could be past, present or future relative to seeing (i.e main clause reference time).
- And just for fun, consider past under present... *George will claim that Mary hit Bill.*
  - Reading 1: hitting is past of utterance time (therefore past of main clause reference time).
  - Reading 2: hitting is future of utterance time, but past of main clause reference time.

SOME RIGHTS RESERVED

Sequence of Tense

- *George will see the woman who is laughing.*
  - How many readings?
- Note that in all of the above cases, the predicate in the subordinate clause describes an event that is extensive in time. Consider readings when subordinate event is punctual.
- *George said that Mary fell.*
  - Falling must precede George's saying.
- *George saw the woman who fell.*
  - Same three readings as before: falling must be past of utterance time, but could be past, present or future relative to seeing (i.e main clause reference time).
- And just for fun, consider past under present… *George will claim that Mary hit Bill.*
  - Reading 1: hitting is past of utterance time (therefore past of main clause reference time).
  - Reading 2: hitting is future of utterance time, but past of main clause reference time.
- And finally…
  1. *A week ago, John decided that in ten days at breakfast he would tell his mother that they were having their last meal together.* (Abusch 1988)
  2. *John said a week ago that in ten days he would buy a fish that was still alive.* (Ogihara 1996)

SOME RIGHTS RESERVED

Interpreting tense in Discourse

- **Example 0.6 (Ordering and Overlap).** *A man walked into the bar. He sat down and ordered a beer. He was wearing a nice jacket and expensive shoes, but he asked me if I could spare a buck.*
- **Example 0.7 (Tense as anaphora?).**
  1. Said while driving down the NJ turnpike: *I forgot to turn off the stove.*
  2. *I didn't turn off the stove.*

SOME RIGHTS RESERVED

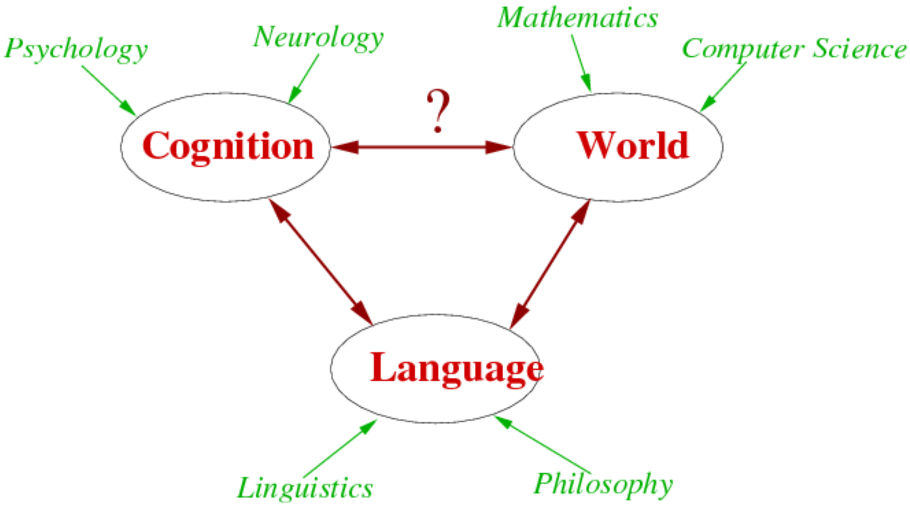
Chapter 15
Conclusion

SOME RIGHTS RESERVED

15.1 A Recap in Diagrams

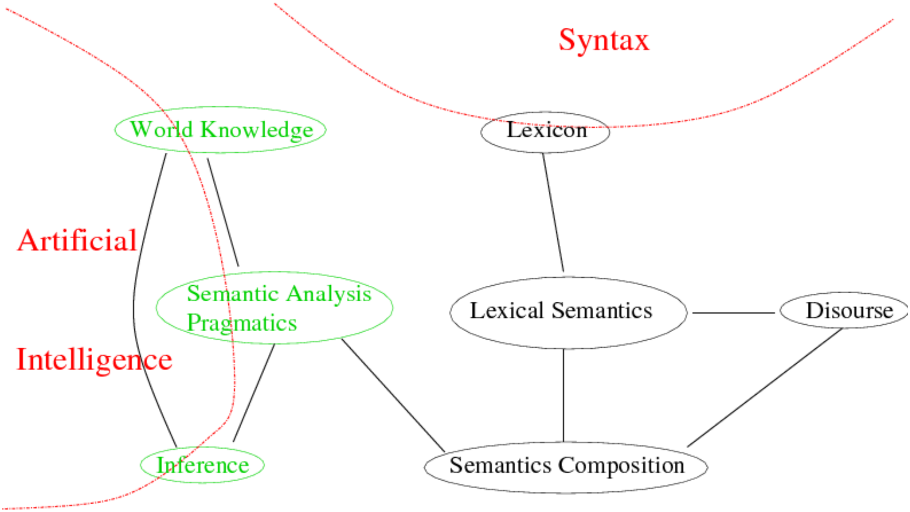
SOME RIGHTS RESERVED

NL Semantics as an Intersective Discipline



SOME RIGHTS RESERVED

A landscape of formal semantics

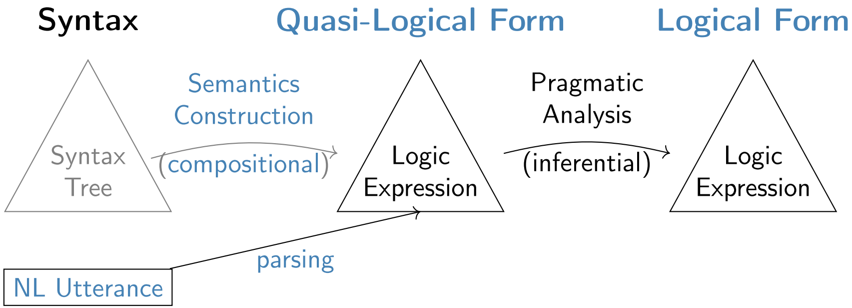


Modeling Natural Language Semantics

- **Problem:** Find formal (logic) system for the meaning of natural language.
- History of ideas
  - Propositional logic [ancient Greeks like Aristotle]
    * *Every human is mortal*
  - First-Order Predicate logic [Frege $\leq$ 1900]
    * *I believe, that my audience already knows this.*
  - Modal logic [Lewis18, Kripke65]
    * *A man sleeps. He snores.* $((\exists X.\mathrm{man}(X) \wedge \mathrm{sleeps}(X))) \wedge \mathrm{snores}(X)$
  - Various dynamic approaches (e.g. DRT, DPL)
    * *Most men wear black*
  - Higher-order Logic, e.g. generalized quantifiers
  - ...

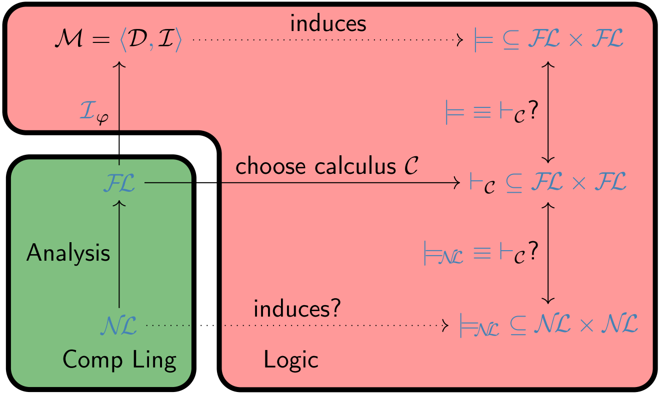
SOME RIGHTS RESERVED

A Semantic Processing Pipeline based on LF



SOME RIGHTS RESERVED

Natural Language Semantics?



SOME RIGHTS RESERVED

15.2 Where to From Here

Where to from here?

- We can continue the exploration of semantics in two different ways:
  - Look around for additional logical/formal systems and see how they can be applied to various linguistic problems. (the logician's approach)
  - Look around for additional linguistic forms and wonder about their truth conditions, their logical forms, and how to represent them. (the linguist's approach)
- Here are some possibilities...

SOME RIGHTS RESERVED

Semantics of Plurals

1. *The dogs were barking.*
2. *Fido and Chester were barking.* (What kind of an object do the subject NPs denote?)
3. *Fido and Chester were barking. They were hungry.*
4. *Jane and George came to see me. She was upset.* (Sometimes we need to look inside a plural!)
5. *Jane and George have two children.* (Each? Or together?)
6. *Jane and George got married.* (To each other? Or to other people?)
7. *Jane and George met.* (The predicate makes a difference to how we interpret the plural)

SOME RIGHTS RESERVED

Reciprocals

- What's required to make these true?
  1. *The men all shook hands with one another.*
  2. *The boys are all sitting next to one another on the fence.*
  3. *The students all learn from each other.*

SOME RIGHTS RESERVED

Presuppositional expressions

- What are presuppositions?
- What expressions give rise to presuppositions?
- Are all apparent presuppositions really the same thing?
  1. *The window in that office is open.*
  2. *The window in that office isn't open.*
  3. *George knows that Jane is in town.*
  4. *George doesn't know that Jane is in town.*
  5. *It was / wasn't George who upset Jane.*
  6. *Jane stopped / didn't stop laughing.*
  7. *George is / isn't late.*

SOME RIGHTS RESERVED

Presupposition projection

1. *George doesn't know that Jane is in town.*
2. *Either Jane isn't in town or George doesn't know that she is.*
3. *If Jane is in town, then George doesn't know that she is.*
4. *Henry believes that George knows that Jane is in town.*

SOME RIGHTS RESERVED

Conditionals

- What are the truth conditions of conditionals?
  1. *If Jane goes to the game, George will go.*
     - Intuitively, not made true by falsity of the antecedent or truth of consequent independent of antecedent.
  2. *If John is late, he must have missed the bus.*
- Generally agreed that conditionals are modal in nature. Note presence of modal in consequent of each conditional above.

SOME RIGHTS RESERVED

Counterfactual conditionals

- And what about these??
  1. *If kangaroos didn't have tails, they'd topple over.* (David Lewis)
  2. *If Woodward and Bernstein hadn't got on the Watergate trail, Nixon might never have been caught.*
  3. *If Woodward and Bernstein hadn't got on the Watergate trail, Nixon would have been caught by someone else.*
- Counterfactuals undoubtedly modal, as their evaluation depends on which alternative world you put yourself in.

SOME RIGHTS RESERVED

Before and after

- These seem easy. But modality creeps in again...
  1. *Jane gave up linguistics after she finished her dissertation.* (Did she finish?)
  2. *Jane gave up linguistics before she finished her dissertation.* (Did she finish? Did she start?)

SOME RIGHTS RESERVED

References I

[Ari10] Mira Ariel. *Defining Pragmatics*. Research Surveys in Linguistics. Cambridge University Press, 2010.

[BB05] Patrick Blackburn and Johan Bos. *Representation and Inference for Natural Language. A First Course in Computational Semantics*. CSLI, 2005.

[Ben91] Johan van Benthem. *Language in Action, Categories, Lambdas and Dynamic Logic*. Vol. 130. Studies in Logic and Foundation of Mathematics. North Holland, 1991.

[Bir13] Betty J. Birner. *Introduction to Pragmatics*. Wiley-Blackwell, 2013.

[Bla+01] Patrick Blackburn et al. "Inference and Computational Semantics". In: *Computing Meaning (Volume 2)*. Ed. by Harry Bunt et al. Kluwer Academic Publishers, 2001, pp. 11–28.

[BRV01] Patrick Blackburn, Maarten de Rijke, and Yde Venema. *Modal Logic*. New York, NY, USA: Cambridge University Press, 2001. isbn: 0-521-80200-8.

[Cho65a] Noam Chomsky. *Aspects of the Theory of Syntax*. MIT Press, 1965.

SOME RIGHTS RESERVED

References II

[Cho65b] Noam Chomsky. *Syntactic structures*. Den Haag: Mouton, 1965.

[Chu40] Alonzo Church. "A Formulation of the Simple Theory of Types". In: *Journal of Symbolic Logic* 5 (1940), pp. 56–68.

[CKG09] Ronnie Cann, Ruth Kempson, and Eleni Gregoromichelaki. *Semantics – An Introduction to Meaning in Language*. Cambridge University Press, 2009. isbn: 0521819628.

[Cre82] M. J. Cresswell. "The Autonomy of Semantics". In: *Processes, Beliefs, and Questions: Essays on Formal Semantics of Natural Language and Natural Language Processing*. Ed. by Stanley Peters and Esa Saarinen. Springer, 1982, pp. 69–86. doi: 10.1007/978-94-015-7668-0_2.

[Cru11] Alan Cruse. *Meaning in Language: An Introduction to Semantics and Pragmatics*. Oxford Textbooks in Linguistics. 2011.

[Dav67a] Donald Davidson. "The logical form of action sentences". In: *The logic of decision and action*. Ed. by N. Rescher. Pittsburgh: Pittsburgh University Press, 1967, pp. 81–95.

[Dav67b] Donald Davidson. "Truth and Meaning". In: *Synthese* 17 (1967).

SOME RIGHTS RESERVED

References III

[de 95] Manuel de Vega. “Backward updating of mental models during continuous reading of narratives”. In: *Journal of Experimental Psychology: Learning, Memory, and Cognition* 21 (1995), pp. 373–385.

[EU10] Jan van Eijck and Christina Unger. *Computational Semantics with Functional Programming*. Cambridge University Press, 2010.

[Fre92] Gottlob Frege. “Über Sinn und Bedeutung”. In: *Zeitschrift für Philosophie und philosophische Kritik* 100 (1892), pp. 25–50.

[GF] *GF - Grammatical Framework*. url: `http://www.grammaticalframework.org` (visited on 09/27/2017).

[GML87] A. M. Glenberg, M. Meyer, and K. Lindem. “Mental models contribute to foregrounding during text comprehension”. In: *Journal of Memory and Language* 26 (1987), pp. 69–83.

[Göd32] Kurt Gödel. “Zum Intuitionistischen Aussagenkalkül”. In: *Anzeiger der Akademie der Wissenschaften in Wien* 69 (1932), pp. 65–66.

SOME RIGHTS RESERVED

References IV

[Har84] D. Harel. "Dynamic Logic". In: *Handbook of Philosophical Logic*. Ed. by D. Gabbay and F. Günthner. Vol. 2. Reidel, Dordrecht, 1984, pp. 497–604.

[HC84] G. E. Hughes and M. M. Cresswell. *A companion to Modal Logic*. University Paperbacks. Methuen, 1984.

[HHS07] James R. Hurford, Brendan Heasley, and Michael B. Smith. *Semantics: A coursebook*. 2nd. Cambridge University Press, 2007.

[Isr93] David J. Israel. "The Very Idea of Dynamic Semantics". In: *Proceedings of the Ninth Amsterdam Colloquium*. 1993. url: `https://arxiv.org/pdf/cmp-lg/9406026.pdf`.

[Jac83] Ray Jackendoff. *Semantics and Cognition*. MIT Press, 1983.

[JL83] P. N. Johnson-Laird. *Mental Models*. Cambridge University Press, 1983.

[JLB91] P. N. Johnson-Laird and Ruth M. J. Byrne. *Deduction*. Lawrence Erlbaum Associates Publishers, 1991.

[Kea11] Kate Kearns. *Semantics*. 2nd. Palgrave Macmillan, 2011.

SOME RIGHTS RESERVED

References V

[Kon04] Karsten Konrad. *Model Generation for Natural Language Interpretation and Analysis*. Vol. 2953. LNCS. Springer, 2004. isbn: 3-540-21069-5. doi: 10.1007/b95744.

[Kri63] Saul Kripke. "Semantical Considerations on Modal Logic". In: *Acta Philosophica Fennica* (1963), pp. 83–94.

[Lee02] Lillian Lee. "Fast context-free grammar parsing requires fast Boolean matrix multiplication". In: *Journal of the ACM* 49.1 (2002), pp. 1–15.

[Lew18] Clarence Irving Lewis. *A Survey of Symbolic Logic*. University of California Press, 1918. url: http://hdl.handle.net/2027/hvd.32044014355028.

[MBV95] Reinhard Muskens, Johan van Benthem, and Albert Visser. "Dynamics". In: ed. by Johan van Benthem and Ter Meulen. Elsevier Science B.V., 1995.

[MR98] C. Monz and M. de Rijke. "A Resolution Calculus for Dynamic Semantics". In: *Logics in Artificial Intelligence. European Workshop JELIA '98*. LNAI 1489. Springer Verlag, 1998.

SOME RIGHTS RESERVED

References VI

[Ohl88] Hans Jürgen Ohlbach. "A Resolution Calculus for Modal Logics". PhD thesis. Universität Kaiserslautern, 1988.

[Par90] Terence Parsons. *Events in the Semantics of English: A Study in Subatomic Semantics*. Vol. 19. Current Studies in Linguistics. MIT Press, 1990.

[Por04] Paul Portner. *What is Meaning? Fundamentals of Formal Semantics*. Blackwell, 2004.

[Pra76] V. Pratt. "Semantical considerations of Floyd-Hoare logic". In: *Proceedings of the 17th Symposium on Foundations of Computer Science*. 1976, pp. 109–121.

[Ran04] Aarne Ranta. "Grammatical Framework — A Type-Theoretical Grammar Formalism". In: *Journal of Functional Programming* 14.2 (2004), pp. 145–189.

[Ran11] Aarne Ranta. *Grammatical Framework: Programming with Multilingual Grammars*. CSLI Publications, 2011. isbn: 1-57586-626-9.

SOME RIGHTS RESERVED

References VII

[RG94] Uwe Reyle and Dov M. Gabbay. “Direct Deductive computation on Discourse Representation Structures”. In: *Linguistics & Philosophy* 17 (1994), pp. 343–390.

[Rie10] Nick Riemer. *Introducing Semantics*. Cambridge Introductions to Language and Linguistics. Cambridge University Press, 2010.

[Rus91] Stuart J. Russell. “An Architecture for Bounded Rationality”. In: *SIGART Bulletin* 2.4 (1991), pp. 146–150.

[Sae03] John I. Saeed. *Semantics*. 2nd. Blackwell, 2003.

[Sau93] Werner Saurer. “A Natural Deduction System for Discourse Representation Theory”. In: *Journal of Philosophical Logic* 22 (1993).

[Sch20] Jan Frederik Schaefer. “Prototyping NLU Pipelines – A Type-Theoretical Framework”. Master’s Thesis. Informatik, FAU Erlangen-Nürnberg, 2020. url: `https://gl.kwarc.info/supervision/MSc-archive/blob/master/2020/Schaefer_Jan_Frederik.pdf`.

SOME RIGHTS RESERVED

References VIII

[Sin94] M. Singer. “Discourse Inference Processes”. In: *Handbook of Psycholinguistics*. Ed. by M. A. Gernsbacher. Academic Press, 1994, pp. 479–515.

[Sta14] Robert Stalnaker. *Context*. Oxford University Press, 2014.

[Ven57] Zeno Vendler. “Verbs and times”. In: *Philosophical Review* 56 (1957), pp. 143–160.

[ZR98] R. A. Zwaan and G. A. Radvansky. “Situation models in language comprehension and memory”. In: *Psychological Bulletin* 123 (1998), pp. 162–185.

[ZS13] Thomas Ede Zimmermann and Wolfgang Sternefeld. *Introduction to Semantics*. de Gruyter Mouton, 2013.

SOME RIGHTS RESERVED