0. What is logic?

But first: What is this class?

See the syllabus!!! See Canvas page! Enroll in **Carnap** page!

- Download textbook: *forallX Fall 2022 MIT edition* (updated throughout semester, but hopefully not too drastically!)
- Problem Sets typically due Fridays 5pm
- Most problem sets submitted online via **Carnap**
- Office hours could change based on when various reading groups are scheduled
- Located on 9th floor, Dreyfoos Wing; turn RIGHT after entering Phil dept

And second: Why come to class?

- We’ll cover the essential content, in a digestible manner (helpful if you find it hard to make time to read)
- We’ll do practice problems! Often geared toward homework and exam questions
- You might gain some ‘insider information’

- This is a **skills-based** course:
  the goal is to learn how to **do** logic, rather than to memorize or regurgitate facts about logic.
- Doing ample practice problems will be **essential**!

Names and Quick Intros

- Ideally you'll feel completely comfortable in this class
- It'll probably be easier to focus if you feel comfortable
- I find it helps to say some words to feel comfortable!
- Let's say some words!

0. What is logic?

a. Arguments and validity

Causes vs. (normative) Reasons for belief

- **Causes of belief**: why you actually believe something
  - Attacked by cat at age 8: causes you to believe ‘cats are dangerous’
  - These are *descriptive* reasons for belief
  - Often these are not rationally compelling reasons
- **Reasons for belief**: why you *ought* to believe something
  - Empirical data on cat attacks per capita
  - These are *normative* reasons for belief
  - Logic aims to characterize the structure of such reasons, when organized into *arguments*

Rhetorically vs. Logically Good Arguments

- **Rhetorically good argument**: argument that actually persuades listeners
  - Might be *descriptively good*, while being logically awful
  - Might rely solely on emotional tricks or rhetoric
  - Anecdotes: “my friend got a blood clot from the Johnson & Johnson Covid vaccine. Therefore, avoid this vaccine!”
  - A plane recently crashed: I better drive!
- **Logically good arguments**: arguments that *ought* to persuade listeners, if they were rational
  - Such arguments might be descriptively pretty unpersuasive!
  - Comparative analysis of risk of blood clots from Janssen vaccine vs. risk of negative Covid health outcome
  - Comparative analysis of plane crashes vs. car crashes

An easy puzzle

Where does Sanjeev live?

Sanjeev lives in Chicago or in Erie.
Sanjeev doesn’t live in Erie.

A: Obviously, in Chicago.

Arguments and sentences

Argument 1

Sanjeev lives in Chicago or in Erie.
Sanjeev doesn’t live in Erie.
Therefore, Sanjeev lives in Chicago.

- Such an argument consists of (declarative) **sentences**.
- Declarative sentences are the kinds that can be **true** or **false**.
- “Therefore” ($\therefore$) indicates that the last sentence (supposedly) **follows from** the first two.
- The last sentence is called the **conclusion**.
- The others are called the **premises**.

Valid and invalid arguments

Argument 2

Mandy enjoys skiing or hiking (or both).
Mandy doesn’t enjoy hiking.
∴ Mandy enjoys skiing.

Argument 3

Mandy enjoys skiing or hiking (or both).
Mandy enjoys skiing.
∴ Mandy doesn’t enjoy hiking.

What’s the difference?

(Deductive) Validity

Definition

An argument is (deductively) **valid** if there is no case where all its premises are true and the conclusion is false.

Definition

An argument is **invalid** if there is at least one case where all its premises are true and the conclusion is false (i.e., if it is not valid).

Definition

A **case** is some hypothetical scenario that makes each sentence in an argument either true or false.

Argument 2 is valid

Argument 2

Mandy enjoys skiing or hiking.

Mandy doesn’t enjoy hiking.

∴ Mandy enjoys skiing.

Argument 2 is **valid**: whenever the premises are true, the conclusion is also true.

Argument 3 is not valid

Argument 3

Mandy enjoys skiing or hiking.
Mandy enjoys skiing.
∴ Mandy doesn’t enjoy hiking.

Argument 3 is **invalid**: there is a possible case where the premises are true and the conclusion isn’t (Mandy enjoys both skiing and hiking).

A harder puzzle

Where does Sarah live?

Sarah lives in Chicago or Erie.
Amir lives in Chicago unless he enjoys hiking.
If Amir lives in Chicago, Sarah doesn’t.
Neither Sarah nor Amir enjoy hiking.

0. What is logic?

b. Cases and determining validity

Validity

Definition

An argument is **valid** if there is no case where all its premises are true and the conclusion is false.

Definition

An argument is **invalid** if there is at least one case where all its premises are true and the conclusion is false (i.e., if it is not valid).

Cases

Definition

A **case** is some hypothetical scenario that makes each sentence in an argument either true or false.

- E.g., imagine you have a friend, her name is Mandy, she loves hiking but hates skiing.
- That’s a case where “Mandy enjoys hiking or skiing” is true.
- Some cases can be imagined even though they never happen IRL, e.g, “It is raining and the skies are clear.”
- Some things you can’t imagine, e.g., “There is a blizzard but there is no wind.”

Determining validity

- Imagine a case where the conclusion is false.
- Are the premises true? You're done: invalid.
- Otherwise, change or expand the case to make them true (without making the conclusion also true).
- Can't? (Probably) valid.

OR

- Imagine a case where all premises are true.
- Is the conclusion false? You're done: invalid.
- Otherwise, change or expand the case to make it false (without making the premises false).
- Can't? (Probably) valid.

Deductively Valid?

Some rodents have bushy tails.

All squirrels are rodents.

∴ Some squirrels have bushy tails.

- Imagine squirrels evolving so that they no longer have bushy tails. Then conclusion is false.
- But premises still true:
  - Imagine chinchillas still have bushy tails.
  - Imagine also that squirrels have not evolved too much—they are still rodents.

Valid?

All rodents have bushy tails.

All squirrels are rodents.

∴ All squirrels have bushy tails.

- If it were invalid, you'd have a case that makes the conclusion false: some squirrels without bushy tails.
- They would have to be rodents still (otherwise premise 2 false).
- And that would require that they have bushy tails (otherwise premise 1 false).

0. What is logic?

c. Other logical notions

Logical Consistency

Definition

Sentences are (logically) **consistent** if there is a case where they are all true.

- also called ‘jointly possible’ or ‘satisfiable’

Definition

Sentences are (logically) **inconsistent** if there is no case where they are all true.

- also called ‘jointly impossible’ or ‘unsatisfiable’

Consistent?

Some carnivores have bushy tails.

All carnivores are mammals.

No mammals have bushy tails.

- No case makes them all true at the same time, so **inconsistent**.

Valid?

Some carnivores have bushy tails.
All carnivores are mammals.
No mammals have bushy tails.
∴ All birds are carnivores.

- The premises cannot all be true in the same case, so inconsistent.
- So: no case makes all the premises true.
- So also: no case makes the premises true and the conclusion false.
- **Arguments with inconsistent premises are automatically valid**, regardless of what the conclusion is.

Tautology (logically necessary)

Definition

A sentence is a **tautology** if there is no case where it is false.

- also called a 'necessary truth' or 'truth-functionally true'

- If it's snowing, then it's snowing.
- Every fawn is a deer.
- The number 5 is prime.
- It's not the case that I am standing and that I am not standing. ('*Law of Non-Contradiction*')

Tautology

What can you say about an argument where the conclusion is a tautology?

- If the conclusion is a tautology, there is no case where it is false.
- So there is no case where both (1) it is false and (2) the premises of the argument are all true.
- $\Rightarrow$ **Arguments with tautologies as conclusions are automatically valid**, regardless of what the premises are.

Logical equivalence

Definition

Two sentences are (logically) **equivalent** if there is no case where one is true and the other is false.

- What can you say about an argument where one of the premises is equivalent to the conclusion? Is it automatically valid?
- Can you have two equivalent sentences that are inconsistent?

0. What is logic?

d. What are we going to learn? And why?

What is logic?

- **Logic is the science of what follows from what.**
- Sometimes a conclusion follows from the premises, sometimes it does not:
  - Mandy lives in Chicago.
    Everyone who lives in Chicago likes hiking.
    ∴ Mandy likes hiking.
  - Mandy lives in Chicago.
    Everyone who likes hiking lives in Chicago.
    ∴ Mandy likes hiking.
- Logic investigates what makes the first argument **valid** and the second **invalid**.

What is formal logic?

- Studies logical properties of **formal languages** (SL and QL, not English).
  - Logical consequence (what follows from what?)
  - Logical consistency (when do sentences contradict one another?)
- Expressive power (what can be expressed in a given formal language, and how?)
- Formal models (mathematical structures described by formal language)
- Inference and proof systems (how can it be proved that something follows from something else?)
- Meta-logical properties of logical systems

Plan for the course

- Sentential Logic (SL), aka Truth-functional logic
  - Symbolization in the formal language of SL (H, $\lor$, &, $\supset$, $\sim$)
  - Testing for validity: truth-tables and TREES!
  - Proofs in natural deduction
- Quantifier Logic (QL), aka First-order logic
  - More fine-grained symbolization ($E(m, h)$, $\forall$ ‘every’, $\exists$ ‘some’, $=$)
  - Semantics: interpretations
  - Proofs in natural deduction
- Metalogic (sprinkled in Unit 1 as well!):
  - Soundness & Completeness of our deduction systems
  - Compactness
  - Expressive completeness, normal forms

What is logic good for? (Philosophy)

- Logic originates in philosophy (e.g. Aristotle); traditionally considered a sub-discipline of philosophy.
- Valid arguments are critical in philosophical research.
- Formal tools of logic are useful for making various philosophical notions precise, e.g.,
  - Possibility and necessity
  - Time
  - Composition and parthood (mereology)
  - Moral obligation and permissibility
  - Belief and knowledge
- Logic applies to the semantics of natural language (philosophy of language, linguistics).

What is logic good for? (Mathematics)

- Formal logic was developed largely in a quest for the foundations of mathematics (19th century).
- Logical systems provide precise foundational framework for mathematics:
  - Axiomatic systems (e.g, geometry)
  - Algebraic structures (e.g., groups)
  - Set theory (e.g, Zermelo–Fraenkel with Choice)
- Precision
  - Formal language makes claims more precise.
  - Formal structures can point to alternatives, unveil gaps in proofs.
  - Formal proof systems make proofs rigorous.
  - Formal proofs make mechanical **proof checking** and **proof search** possible.

What is logic good for? (Computer Science)

- Computer science deals with lots of formal languages.
- Logic is a good example of how to set up and use formal languages.
- ‘Logic : Computer Science’ = ‘Calculus : Natural Science’
- Applications of logical systems in CS are numerous:
    - Combinational logic circuits
    - Database query languages
    - Logic programming
    - Knowledge representation
    - Automated reasoning
    - Formal specification and verification (of programs, of hardware designs)
    - Theoretical computer science (theory of computational complexity, semantics of programming languages)

0. What is logic?

e. Symbolization and SL

Validity in virtue of form

Argument 1

Sanjeev lives in Chicago or Erie.

Sanjeev doesn't live in Erie.

$\therefore$ Sanjeev lives in Chicago.

Argument 2

Mandy enjoys skiing or hiking.

Mandy doesn't enjoy hiking.

$\therefore$ Mandy enjoys skiing.

Form of arguments 1 & 2

X or Y.

Not Y.

$\therefore X$.

Some valid argument forms

Disjunctive syllogism

X or Y.

Not Y.

$\therefore$ X.

Modus ponens

If X then Y.

X.

$\therefore$ Y.

Hypothetical syllogism

If X then Y.

If Y then Z.

$\therefore$ If X then Z.

Symbolizing arguments

Symbolization key

S: Mandy enjoys skiing

H: Mandy enjoys hiking

Argument 2

Mandy enjoys skiing or Mandy enjoys hiking.	$(S \vee H)$
Not: Mandy enjoy hiking.	$\sim H$
$\therefore$ Mandy enjoys skiing.	$\therefore S$

The language of SL

- CAPITAL **Sentence letters**, such as ‘H’ and ‘S’, to symbolize *atomic sentences* (e.g. ‘Mandy likes hiking’)
- **Connectives**, to indicate how atomic sentences are connected

$\vee$	either . . . or . . .	[‘inclusive or’]
&	both . . . and . . .	
$\supset$	if . . . then . . .	[‘material conditional’]
$\sim$	not . . .	[it is not the case that]

This can get complicated, e.g.:

“Mandy enjoys skiing or hiking, and if she lives in Erie, she doesn’t enjoy both.”

$$((S \vee H) \,\&\, (E \supset \sim(S \,\&\, H)))$$

0. What is logic?

f. Bonus: Some History of Logic!

The beginnings

"The Philosopher"

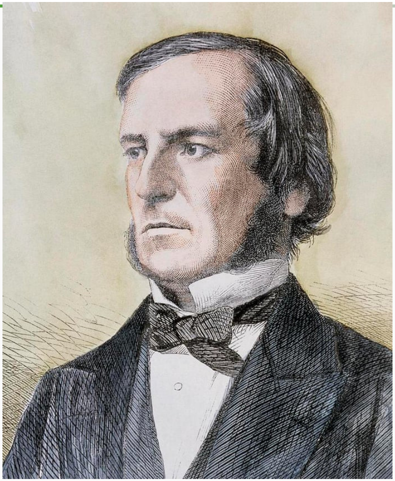
- Rules of debate & rhetoric
- Ancient India: Gautama, **Nyāya Sūtras** (600 BCE–200 CE)
- Ancient Greece: Aristotle (384-322 BCE)
- Cataloged valid arguments ("syllogisms"), e.g.,
- All ungulates have hooves.
  No fish have hooves.
  ∴ No fish are ungulates.

The middle ages

Avicenna!

- Ibn Sīnā (Avicenna) (980-1037): worked on like *everything*
- Pierre Abelard (1079-1142)
- William Ockham (1285-1347)
- Jean Buridan (1301-1358)

Mathematical logic


Boole!

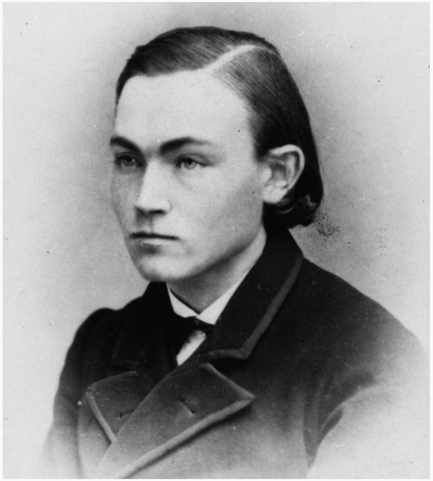
- Augustus De Morgan (1806-1871)
- George Boole (1815-1864) [self–taught, and so can you!]
- Charles Lutwidge Dodgson (aka Lewis Caroll) (1832-1898) [thank him for the **trees!**]
- John Venn (1834-1923)

Modern logic: Peirce at al

Ladd Franklin!

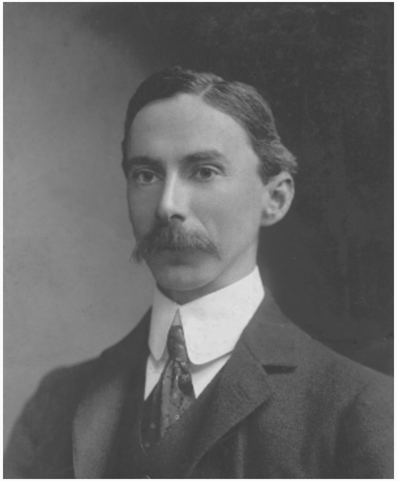
- Charles Sanders Peirce (1839-1914): a Cambridge native!
- Christine Ladd Franklin (1847-1930)
- Ernst Schröder (1841-1902)

Modern logic: Gottlob Frege



- 1848-1925
- Predicates and quantifiers
- Plan to turn all of math into theorems of logic alone
- Did Frege plagarize ideas from the Stoics???

Modern logic: Bertrand Russell



- 1870-1972
- Showed Frege's system contradictory (1902)
- Fixed it (*Principia mathematica* 1910-13, 3 vols.)
- Plan to turn all of math into theorems of logic alone

Modern logic: David Hilbert

- 1862-1943
- Combined Russell’s and Schröder’s systems
- First modern logic textbook
- Plan to turn all of math into consequences of a single set of premises

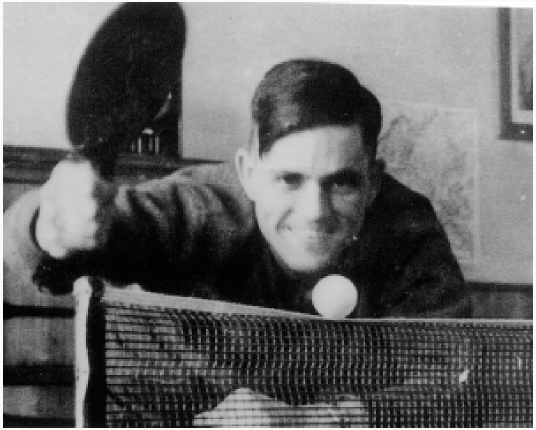
Modern logic: Kurt Gödel

- 1906-1978
- Showed that every valid argument has a proof
- Showed that Frege/Russell's and Hilbert's plans can't work

Modern logic: Alan Turing

- 1912-1954
- Showed that unlike SL, QL has no decision procedure
- Invented Turing machines ("father of computer science")

Modern logic: Gerhard Gentzen


Ping pong let’s GOOOOOOO!

- 1909-1945
- Invented natural deduction
- Founded theory of proofs

Modern logic: modal logic

Barcan Marcus!

- Extend logic with operators for "possible" and "necessary"
- Pioneered by philosophers, now used by computer scientists
- Rudolf Carnap, Saul Kripke, Ruth Barcan Marcus

1. Symbolization in SL

1. Symbolization in SL

a. Symbolization keys and paraphrase

Symbolizing arguments

Argument 2 (Paraphrased)

Mandy enjoys skiing or Mandy enjoys hiking.

Not: Mandy enjoy hiking.

$\therefore$ Mandy enjoys skiing.

Form of argument 2

S or H.

Not H.

$\therefore S$.

Symbolization of argument 2 in SL

$(S \vee H)$

$\sim H$

$\therefore S$

Symbolization keys

Definition

A symbolization key is a list that pairs **sentence letters** with the basic English sentences they represent.

For instance:

Symbolization key

S: Mandy enjoys skiing
H: Mandy enjoys hiking

Symbolization keys

- Sentence letters are UPPERCASE, possibly with subscripts (e.g., H_1, H_2).
- Usually the symbolization key is given to you (to facilitate grading ;)
- It should not be possible to further decompose the “atomic sentences” represented by sentence letters.

  For instance:

  B: Mandy enjoys skiing or hiking

  is a bad choice of symbolization.

Paraphrase

- Successful symbolization sometimes requires **paraphrase** to ensure atomic sentences appear explicitly.
- Two things to watch for: pronouns and coordination.
- Pronouns stand in for singular terms (e.g., names): explicitly replace pronouns by those names.
- “or”, “both . . . and”, “neither . . . nor” can connect sentences but also noun phrases and verb phrases:
  paraphrase them so that they connect sentences.

Pronouns

Example

If Mandy enjoys hiking, **she** also enjoys skiing.

Replace “she” by “Mandy”:
If [Mandy enjoys hiking] then [Mandy enjoys skiing].

Coordination of noun phrases

Example

Mandy and Sanjeev enjoy hiking.

Both [Mandy enjoys hiking] and [Sanjeev enjoys hiking].

Example

Sanjeev lives in **Erie or Chicago**.

Either [Sanjeev lives in Erie] or [Sanjeev lives in Chicago].

Exercise caution!

Good

Mandy and Sanjeev ate pizza.

Both [Mandy ate pizza] and [Sanjeev ate pizza].

Bad

Mandy and Sanjeev ate the whole pizza.

Both [Mandy ate the whole pizza] and [Sanjeev ate the whole pizza].

Coordination of verb phrases

Example

Mandy enjoys **skiing or hiking**.

Either [Mandy enjoys skiing] or [Mandy enjoys hiking].

Example

If Sanjeev enjoys **skiing and hiking**, he lives in Chicago.

If [Sanjeev enjoys skiing] and [Sanjeev enjoys hiking], then [Sanjeev lives in Chicago].

1. Symbolization in SL

b. Basic symbolization

Negation

- **Paraphrase** grammatical negation ("is not", "does not") using the corresponding atomic sentence prefixed by "**it is not the case that**."
- **Symbolize** "it is not the case that A" as $\sim A$.

Example

Mandy **doesn't** enjoy skiing.
It is not the case that [**Mandy enjoys skiing**].
It is not the case that S.
$\sim S$

Conjunction

- **Paraphrase** sentences connected by “and”, “but”, “even though”, “yet”, and “although” using “**both *A* and *B***”
- **Symbolize** “both *A* and *B*” as $(A \& B)$.

Example

Even though Mandy lives in Erie**,** she enjoys hiking.
Both [**Mandy lives in Erie**] and [**Mandy enjoys hiking**].
Both *E* **and** *H*.
$(E \& H)$

Disjunction (it's inclusive!)

- **Paraphrase** sentences connected by "or" using "**either** A **or** B"
- **Symbolize** "either A or B" as $(A \lor B)$.

Example

Sanjeev lives in Chicago **or** Erie.
Either [**Sanjeev lives in Chicago**] or [**Sanjeev lives in Erie**].
Either C **or** E.
$(C \lor E)$

Ignore the suggestion that "either . . . or . . . " is exclusive. We'll always treat it as inclusive unless explicitly stated.

Conditional (it’s material!)

- **Paraphrase** using “**if A then B**” any sentence of the form:
  - “if A, B”
  - “B if A” (note order is reversed with lonely if!)
  - “B provided A”; and also “B given A”
- **Symbolize** “if A then B” as $(A \supset B)$.

  note our funny ‘horseshoe’ symbol ‘$\supset$’.

  Round ‘em up! (On *Carnap* use ‘>’)

Example

- Mandy enjoys hiking **if** Sanjeev lives in Chicago.
- If [**Sanjeev lives in Chicago**] then [**Mandy enjoys hiking**].
- **If** C **then** H.
- $(C \supset H)$

The parts of a conditional

- $(A \supset B)$ symbolizes:
  - "if A, B"
  - "B if A" (note order is reversed!)
  - "B provided A"
- A is the **antecedent**: it symbolizes the condition that has to be met for the "then" part to apply. (Like a promise!)
- B is the **consequent**: it symbolizes what must be true (e.g. to keep your promise!) if the antecedent condition is true.

Mix & match

Example

Mandy doesn't enjoy hiking, **provided** Sanjeev lives in Chicago or Erie.

If [Sanjeev lives in Chicago **or** Erie] then [Mandy **doesn't** enjoy hiking].

If (either [**Sanjeev lives in Chicago**] or [**Sanjeev lives in Erie**]) then (it is not the case that [**Mandy enjoys hiking**]).

If (either C or E) then (it is not the case that H).

$((C \vee E) \supset {\sim}H)$

1. Symbolization in SL

c. Conditionals

A logic puzzle

- Every card has a letter on one side and a number on the other side.

- You're a card inspector tasked with making sure that cards satisfy this quality standard:

  If a card has an even number on one side, then it has a vowel on the other.

A logic puzzle

Which card(s) do you have to turn over to make sure that:

> *If a card has an even number on one side, then it has a vowel on the other.*

E	K	3	4
(1)	(2)	(3)	(4)

Another logic puzzle

- At an all-ages event where everyone has a drink
- You know how old some of the people are, and you can tell what some of them are drinking
- You're tasked with making sure that the following rule is followed:

  If a person is drinking alcohol, then they are at least 21 years old.

Another logic puzzle

Which of these people do you have to check (age or drink) to ensure that:

> *If a person is drinking alcohol, then they must be at least 21 years old.*

22 years	16 years	drinks pop	drinks beer
(1)	(2)	(3)	(4)

Truth conditions of Material conditionals

$$\text{If } \underbrace{\text{X is drinking alcohol}}_{A}, \text{ then } \underbrace{\text{X is over 21}}_{B}$$

- "If A, then B" can only be **false** if:
  - A is **true**: we check age if X is drinking beer (A true), not if drinking pop; **and**
  - B is **false**: we check drink if X underage (B false), not if over 21 (B true)
- "If A, then B" is true if:
  - A is **false** (we don't check people drinking pop); **or**
  - B is **true** (those 21+ can drink whatever they want!);
  - (or both)

Understanding material conditionals as promises

- A promise: *if Sanjeev has to go to the airport, then I will drive him*
- Notice that you *vacuously* keep your promise if Sanjeev never goes to the airport
- You can't break a promise whose conditions are not satisfied

- The material conditional is just like this: it is *vacuously* true whenever the antecedent is false
- You'll have to get used to this!
- Other conditionals (e.g. causal, subjunctive, counterfactual) are not *truth-functional* (so handling them is controversial!)

1. Symbolization in SL

d. “Only if” and “unless”

‘only if’ vs. a lonely ‘if’

- Sue drinks beer (A) **only if** she is over 21 (B)

$$A \supset B$$

- False if Sue drinks beer (A), but is underage ($\sim B$)
- Sue drinks beer (A), if she is over 21 (B).

$$B \supset A$$

- False if she’s 25 (B), but drinks pop ($\sim A$).
- Not false if she’s 16 and drinking beer.

Conditional recap: only if you can't remember!

- **Paraphrase** "A only if B" as "**if A then B**".
- **Symbolize** "A only if B" as $(A \supset B)$.
- Note:
  - "A if B" is $(B \supset A)$ (lonely if)
  - "A only if B" is $(A \supset B)$ (same order as 'if P, then Q')
- We'll come back to the *biconditional*:

  Symbolize "A if and only if B" as $(A \equiv B)$.

Conditional Practice: only if vs. lonely if

Schematize the following sentence:

Jack will go to the store only if Susie goes to the store, and Susie will go if Beatrice goes.

Symbolization Key:

- J: Jack will go to the store
- S: Susie goes to the store
- B: Beatrice goes to the store

Answer: $(J \supset S) \& (B \supset S)$

Unless (confusing unless you use a trick!)

Which of these people do you have to check (age or drink) to ensure that:

People are drinking pop unless they are over 21.

22 years	16 years	drinks pop	drinks beer
(1)	(2)	(3)	(4)

Unless

$$\underbrace{\text{X is drinking pop}}_{A}\text{, unless }\underbrace{\text{X is over 21}}_{B}$$

- "*A* unless *B*" can only be **false** if:
  - *A* is **false** (we check age if person is drinking beer), **and**
  - *B* is **false** (we check drink if person not at least 21)
- "*A* unless *B*" is true (test OK) if *A* or *B* or both are true.
- "*A* unless *B*" can be paraphrased and symbolized by:
  - "*A* if not *B*" ($\sim B \supset A$)
  - "either *A* or *B*" ($A \vee B$) [**Remember this one**!!!]

Trick for handling ‘Unless’

Treat “**unless**” the same way you would treat “or”

Example

Mandy enjoys hiking unless Sanjeev lives in Chicago.

$(H \vee C)$

- Since disjunction is symmetric, you won’t have to remember order of atomic sentences if you symbolize ‘unless’ using ‘or’.
- So you can’t go wrong with this approach!

1. Symbolization in SL

e. More connectives

Biconditional: If and only if (‘$\equiv$’)

Example

Mandy enjoys hiking **if and only if** she enjoys skiing.
Both [if S then H] and [if H then S].
$((S \supset H) \& (H \supset S))$
$(H \equiv S)$

Unlike the conditional, the biconditional is symmetric: the order of atomic sentences does not matter!

Exclusive or

Paraphrase sentences containing “either A or B but not both” using “**both [either A or B] and**
[it is not the case that [both A and B]]”

Example

Mandy enjoys hiking or skiing but not both.

Both [either H or S] and
[it is not the case that [both H and S]].

$((H \vee S) \mathbin{\&} {\sim}(H \mathbin{\&} S))$

An alternative: $(H \mathbin{\&} {\sim}S) \vee (S \mathbin{\&} {\sim}H)$
i.e. Either (H and not S) or (S and not H)

Neither ... nor ... ('nand')

Paraphrase sentences containing "neither *A* nor *B*" using
"**both [it is not the case that *A*] and**
[it is not the case that *B*]"

Example

Mandy enjoys neither hiking nor skiing.

Both [it is not the case that *H*] and
[it is not the case that *S*].

$(\sim H \,\&\, \sim S)$

This connective has a special metalogical property that we might discuss at some point!

Mix & match

Example

Sarah lives in Chicago or Erie.

Either [Sarah lives in Chicago] or [Sarah lives in Erie].

Amir lives in Chicago unless he enjoys hiking.

Either [Amir lives in Chicago] or [Amir enjoys hiking].

If Amir lives in Chicago, Sarah doesn't.

If [Amir lives in Chicago] then [it is not the case that [Sarah lives in Chicago]].

Neither Sarah nor Amir enjoy hiking.

Both [it is not the case that [Sarah enjoys hiking]] and [it is not the case that [Amir enjoys hiking]].

∴ Sarah lives in Erie.

Mix & match

Example

Sarah lives in Chicago or Erie.

$(C \vee E)$

Amir lives in Chicago unless he enjoys hiking.

$(A \vee M)$

If Amir lives in Chicago, Sarah doesn't.

$(A \supset {\sim}C)$

Neither Sarah nor Amir enjoy hiking.

$({\sim}S \,\&\, {\sim}M)$

$\therefore$ Sarah lives in Erie.

$\therefore\ E$.

1. Symbolization in SL

f. Defining Formulae in SL

Will the real SL please stand up?

What we have said so far about Sentential Logic (SL):

- Uppercase letters, subscripts allowed: e.g. J, H, B, N_3
- Punctuation: parentheses '(' and ')' ;(no brackets or braces!)
- Connectives: $\sim$, & , $\vee$, $\supset$, $\equiv$
- But so is any string of these symbols a legitimate formula of SL?
- e.g. $((((A \,\&\, \sim(B \supset C \equiv D((($

Expressions vs. Well-formed Formula (WFFs)

- **Expression of SL**: any finite string of symbols from language SL
  - Includes ridiculous-seeming strings, e.g. $(((((())LOGIC_1FOREVER((($
- **Well-formed Formulae** (WFFs): often shortened to ‘formulae’ or ‘**sentences**’ of SL
  - These are the expressions we are interested in
  - They end up being expressions that are true or false under an assignment of truth values to atomic sentence letters
- But how do we rigorously define the WFFs as a subset of the expressions?

Look it's a bird! It's a plane! It's....RECURSION!

We define the well-formed formulae (wffs) *recursively*:

1. Each atomic sentence is a wff (base case)
2. If $\mathcal{P}$ is a wff, then so is $\sim\mathcal{P}$
3. If $\mathcal{P}$ and $\mathcal{Q}$ are both wffs, then so are:
   - $(\mathcal{P} \,\&\, \mathcal{Q})$
   - $(\mathcal{P} \vee \mathcal{Q})$
   - $(\mathcal{P} \supset \mathcal{Q})$
   - $(\mathcal{P} \equiv \mathcal{Q})$
4. And that's all folks! (No other expressions of SL are wffs)

Identifying the Main Connective

- Apply the recursive definition in reverse!

- Ask if the sentence is a negation of another wff
- If not, locate the two wffs in the scope of the outermost parentheses, and figure out which connective combines them
- Rinse and repeat: stop when you reach the atomic sentences

- $\big(\sim(A \supset (\sim B \vee C)) \mathbin{\&} ((A \equiv B) \supset C)\big)$
- $\big(\qquad \mathcal{P} \qquad\qquad \& \qquad\qquad \mathcal{Q} \qquad\big)$

But wait? What are these '$\mathcal{P}$'s and '$\mathcal{Q}$'s

- $\mathcal{P}$ is a **metavariable** we use to talk about (to mention!) an arbitrary expression from SL
- The symbol '$\mathcal{P}$' is not itself an expression in SL
- Rather, '$\mathcal{P}$' is part of our **metalanguage**, which in this case is English, augmented by a bunch of symbols
- So technically, '$\sim\mathcal{P}$' is not a sentence of SL! So we define a convention: '$\sim\mathcal{P}$' abbreviates the result of concatenating the sentence $\mathcal{P}$ with the negation symbol '$\sim$'.
- BOOM!
- Who knew that air–quotes ('scare–quotes'?) could play such an important role in the foundations of logic?

1. Symbolization in SL

g. Use vs. ‘Mention’

Use vs. 'Mention'

- When we **use** a word or symbol, it does not have quotation marks:
  - Logic is a fascinating subject!
  - You are sitting through a logic lecture.
  - Logic is a 21st-century American rapper.

- When we **mention** a word or symbol, we use quotation marks:
  - The name of this course is 'Logic I'
  - There is a 21st-century American rapper called 'Logic'
  - The rapper Sir Robert Hall goes by the name 'Logic'
  - We symbolize conjunction as '&' and disjunction using '$\vee$'.

Why this matters: Object vs. Metalanguage

- We talk *about* an object language by *using* a metalanguage
- SL is an object language we talk about using English
- When we talk about SL, we MENTION expressions in SL, and thereby should technically put quotation marks around these:
  - BAD: $P \& Q$ is a sentence in SL
  - **GOOD**: "$P \& Q$" is a sentence in SL
  - We have to mention "$P \& Q$" since it is not a sentence in English; it's a sentence in SL. So we can't *use* it in English.
- But psssst: we'll often be lazy and won't put quotes where we're technically supposed to...

Use vs. ‘Mention’

- Some more detailed glosses, in case one of these clicks:
- When we use a word, it does not have quotation marks
  - The thing referenced by the word is being *put to good use*
  - Barack Obama was the 44th president of the USA
- When we mention a word or string of symbols in a sentence, we put quotation marks around them to show that we are mentioning something:
  - “Obama” names the 44th president of the USA
  - “Obama” names Obama; “Obama” is a name.
  - “ “Obama” ” is a quotation-name (i.e. a name of a name)
  - “$p \,\&\, {\sim} q$” is a conjunction, whose second conjunct is a negation.

A Mistake on the Wiki!

- On a Wiki page discussing the geographical usage of ‘soda’ vs. ‘pop’ in the USA
- Sentence: *To a lesser extent soda is also fairly common further down the east coast in eastern Virginia, eastern Carolinas and coastal Florida. Here, soda is not too dominant but competes with multiple other terms*
- Cite your sources!

Use vs. ‘Mention’ PRACTICE!

Punctuate the following sentences with quotation-marks so as to make them true:

1. Hunt is the last name of the instructor for this course.
2. Hunt is a surname which begins with the letter H.
3. I call my dog Ivy Ivy, but I never use Ivy’s first name in addressing her.

Answers:

1. “Hunt” is the last name of the instructor for this course.
2. “Hunt” is a surname which begins with the letter “H”.
3. I call my dog Ivy “Ivy”, but I never use “Ivy’s first name” in addressing her.

1. Symbolization in SL

h. Ambiguity

Types of ambiguity

- Lexical ambiguity: one word—many meanings
  e.g., “bank”, “crane”

- Syntactic ambiguity: one sentence—many readings
  e.g.,
  - “Flying planes can be dangerous” (Chomsky)
  - “One morning I shot an elephant in my pajamas.
    How he got in my pajamas, I don’t know.” (Groucho Marx)

Ambiguity of & and $\vee$

- English sentences don’t have parentheses.
- This can lead to ambiguity, e.g.,

  Ahmed admires Brit and Cara or Dina.
- It might mean one of:

  Ahmed admires either [both Brit and Cara] or Dina.

  Ahmed admires both Brit and [either Cara or Dina].
- In SL, symbolizations are unambiguous:

  $((B \& C) \vee D)$

  $(B \& (C \vee D))$

The man who was hanged by a comma

- Sir Roger Casement (1864–1916)
- British consul to Congo and Peru
- Tried to recruit Irish revolutionaries in Germany during WWI
- Tried for treason

Treason Act of 1351 (20th century court added the comma!)

ITEM, Whereas divers Opinions have been before this Time in what Case Treason shall be said, and in what not; the King, at the Request of the Lords and of the Commons, hath made a Declaration in the Manner as hereafter followeth, that is to say; When a Man doth compass or imagine the Death of our Lord the King, or of our Lady his Queen or of their eldest Son and Heir; or if a Man do violate the King's Companion, or the King's eldest Daughter unmarried, or the Wife of the King's eldest Son and Heir; or **if a Man** do levy War against our Lord the King in his Realm, or **be adherent to the King's Enemies in his Realm, giving to them Aid and Comfort in the Realm , or elsewhere**, and thereof be probably attainted of open Deed by the People of their Condition: . . . And it is to be understood, that in the Cases above rehearsed, that ought to be judged Treason which extends to our Lord the King, and his Royal Majesty: . . .

R v. Casement in SL

- Symbolization key:

  A: Casement was adherent to the King's enemies in the realm. (false)

  G: Casement gave aid and comfort to the King's enemies in the realm. (false)

  B: Casement was adherent to the King's enemies abroad. (true)

  H: Casement gave aid and comfort to the King's enemies abroad. (false)
- Without the comma, 'treason' defined as:

  $A \vee (G \vee H)$
- With the comma, 'treason' now defined as:

  $(A \vee B) \vee (G \vee H)$ [and B **is** true]

1. Symbolization in SL

i. Practice Problems!

Validity, Paraphrasing, Symbolizing

Classify the following arguments as sound, valid but unsound, or invalid. Next, paraphrase the argument in logical form. Finally, symbolize the argument in SL.

1. Detroit will win the Super Bowl this year if they didn't make the playoffs. Detroit didn't make the playoffs. Therefore: Detroit will win the Super Bowl this year.
2. If Aristotle was a student of Plato, then Aristotle lived in Athens. Aristotle lived in Athens. Hence, Aristotle was Plato's student.
3. Cleveland is a city in Ohio. Pittsburgh is a city in Ohio. Therefore: there are at least two cities in Ohio.
4. Either Toledo is in Michigan or the Upper Peninsula is in Michigan. Toledo is not in Michigan. Therefore: the Upper Peninsula is in Michigan.

Solution to Question 1:

- Valid argument (e.g. of modus ponens—affirming the antecedent) but unsound: the first premise (the conditional) is false
- Paraphrase of this argument:

  If (it is not the case that Detroit made the playoffs) then (Detroit will win the Super Bowl).
  It is not the case that Detroit made the playoffs.
  Therefore, Detroit will win the Super Bowl.
- Symbolization in SL:
  - Let ‘P’ stand for ‘Detroit made the playoffs’
  - Let ‘D’ stand for ‘Detroit will win the Super Bowl’

  $(\sim P) \supset D$
  $\sim P$
  $\therefore D$

More Complex Symbolizations

Paraphrase the following English sentences using our logical symbolism. Any condensed clauses should be expanded into complete sentences. Identify as well the main connective and practice symbolizing with atomic sentences.

1. If they either drain the swamp and reopen the road or dredge the harbor, they will provide the uplanders with a market and themselves with a bustling trade.
2. Unless his new novel does very well and gets him a large advance, Malone will take a position in a college writing program or he will take out a second mortgage and not sell his car.
3. If Germany annexes Austria, Czechoslovakia will be militarily defensible only if France honors her treaty obligations and arranges for the transit of Soviet troops across Poland or Romania.

Drain the Swamp!

Sentence: *If they either drain the swamp and reopen the road or dredge the harbor, they will provide the uplanders with a market and themselves with a bustling trade.*

- Paraphrase: **IF** [(they drain the swamp and they reopen the road) or (they dredge the harbor)] **THEN** (they will provide the uplanders with a market and they will provide themselves with a bustling trade).
- Alternative paraphrase using connectives: [(they drain the swamp & they reopen the road) $\vee$(they dredge the harbor)] $\supset$(they will provide the uplanders with a market & they will provide themselves with a bustling trade)].

Drain the Swamp (still draining...)

Paraphrase: **IF** [(they drain the swamp and they reopen the road) or (they dredge the harbor)] **THEN** (they will provide the uplanders with a market and they will provide themselves with a bustling trade).

Symbolization Key:

S: they drain the swamp
R: they reopen the road
H: they dredge the harbor
M: they will provide the uplanders with a market
T: they will provide themselves with a bustling trade

Symbolization–answer: $((S \& R) \vee H) \supset (M \& T)$

2. SL and truth tables

2. SL and truth tables

a. Characteristic truth tables

Sentence letters and connectives

- Symbolization involves **sentence letters** like H and **connectives** ($\sim$, $\vee$, &, $\supset$, $\equiv$)
- Recall that a **case** makes atomic sentences **true** or **false** (and never both).
- So if we can determine **truth conditions** for sentences involving connectives, we can assign **true** and **false** also to results of symbolization.

When is (H & S) true?

(H & S) is true if and only if H is true and S is also true.
Suppose a case makes H true and S false.
In that case, (H & S) would be **false**.

Negation $\sim$

Definition

$\sim\mathcal{A}$ is true iff $\mathcal{A}$ is false.

Characteristic truth table:

$\mathcal{A}$	$\sim\mathcal{A}$
T	**F**
F	**T**

Conjunction &

Definition

$(\mathcal{A} \& \mathcal{B})$ is true iff $\mathcal{A}$ is true and $\mathcal{B}$ is true, and false otherwise.

Characteristic truth table:

$\mathcal{A}$	$\mathcal{B}$	$(\mathcal{A} \& \mathcal{B})$
T	T	T
T	F	F
F	T	F
F	F	F

Disjunction $\vee$

Definition

$(\mathcal{A} \vee \mathcal{B})$ is true iff $\mathcal{A}$ is true or $\mathcal{B}$ is true (or both), and false otherwise.

Characteristic truth table:

$\mathcal{A}$	$\mathcal{B}$	$(\mathcal{A} \vee \mathcal{B})$
T	**T**	**T**
T	**F**	**T**
F	**T**	**T**
F	**F**	**F**

A logic puzzle

Which card(s) do you have to turn over to make sure that:

If a card has an even number on one side, then it has a vowel on the other.

E	K	3	4
(1)	(2)	(3)	(4)

The material conditional $\supset$

Definition

$(\mathcal{A} \supset \mathcal{B})$ is true iff $\mathcal{A}$ is false or $\mathcal{B}$ is true (or both), and false otherwise.

$\mathcal{A}$	$\mathcal{B}$	$(\mathcal{A} \supset \mathcal{B})$
T	T	T
T	F	F
F	T	T
F	F	T

Memorize 2nd row!: Conditional is false ONLY in case of a counter–example, i.e. case where antecedent is true but consequent is false

The material biconditional $\equiv$

Definition

$(\mathcal{A} \equiv \mathcal{B})$ is true iff $\mathcal{A}$ and $\mathcal{B}$ have the same truth value, and false otherwise.

$\mathcal{A}$	$\mathcal{B}$	$(\mathcal{A} \equiv \mathcal{B})$
T	T	T
T	F	F
F	T	F
F	F	T

2. SL and truth tables

b. Sentences of SL

Sentences of SL (*Well-formed Formulae* (WFFs)

Definition

1. Every sentence letter is a sentence (wff).
2. If $\mathcal{A}$ is a sentence, then $\sim\mathcal{A}$ is a sentence.
3. If $\mathcal{A}$ and $\mathcal{B}$ are sentences, then
   - $(\mathcal{A} \,\&\, \mathcal{B})$ is a sentence.
   - $(\mathcal{A} \vee \mathcal{B})$ is a sentence.
   - $(\mathcal{A} \supset \mathcal{B})$ is a sentence.
   - $(\mathcal{A} \equiv \mathcal{B})$ is a sentence.
4. Nothing else is a sentence.

The indicated connective is called the **main connective**.

Construction of sentences

- H is a sentence.
- S is a sentence.
- $(H \vee S)$ is a sentence.
- $(H \mathbin{\&} S)$ is a sentence.
- $\sim(H \mathbin{\&} S)$ is a sentence.
- $((H \vee S) \mathbin{\&} \sim(H \mathbin{\&} S))$ is a sentence.

(Main connective is **highlighted**.)

Examples of non-sentences (i.e. NOT well-formed formulae)

- *HikesMandy*

  single sentence letters
- $(H \sim S)$

  $\sim$ can't go between sentences
- $(H \,\&\, S \,\&\, C)$

  & combines only two sentences
- $({\sim}H)$

  no parentheses around ${\sim}H$
- $(H \supset (S \,\&\, C)$

  missing closing parenthesis
- $H \vee S$

  missing parentheses
- $[H \supset (S \,\&\, C)]$

  only one kind of parentheses allowed: no brackets!

2. SL and truth tables

c. Truth value assignments (a.k.a. ‘valuations’)

Truth value assignment (a.k.a. ‘valuation’)

Definition

A **truth value assignment** (TVA) (a.k.a. **valuation**) is an assignment of **T** or **F** to each atomic sentence letter in a sentence or sentences.

Definition

The **truth value of a sentence** $\mathcal{S}$ on a valuation is:

1. if $\mathcal{S}$ is a sentence letter: the truth value assigned to it
2. if $\mathcal{S}$ is $\sim\mathcal{A}$: opposite of the truth value of $\mathcal{A}$
3. if $\mathcal{S}$ is $(\mathcal{A} * \mathcal{B})$: result of characteristic truth table of $*$ for truth values of $\mathcal{A}$ and $\mathcal{B}$.

Computing truth values

Truth-value assignment (TVA): H is **T**, S is **F**.

On this valuation:

- H is **T**.
- S is **F**.
- $(H \vee S)$ is **T** (because '**T** $\vee$ **F**' gives **T**).
- $(H \,\&\, S)$ is **F** (because '**T** & **F**' gives **F**).
- $\sim(H \,\&\, S)$ is **T** (because $\sim$**F** is **T**).
- $((H \vee S) \,\&\, \sim(H \,\&\, S))$ is **T** (because '**T** & **T**' gives **T**).

Computing truth values

H	S	$((H$	$\vee$	$S)$	$\&$	$\sim$	$(H$	$\&$	$S))$
T	**F**	**T**	**T**	**F**	**T**	**T**	**T**	**F**	**F**
					$\uparrow$				

- Copy truth values under atomic sentence letters.
- Compute values of parts that combine sentence letters (working from 'inside out' i.e. from minor connectives toward the main connective)
- Use computed values for larger parts.
- Done row when you have the value under the main connective.
- Complete this process for each row (each TVA)

2. SL and truth tables

d. Validity and truth tables

Validity

- Recall: an argument is (deductively) **valid** if there is no **case** where all premises are true and the conclusion is false.
- In a **case**, every **atomic sentence** is either true or false (but not both!).
- In SL, **valuations** make every **atomic sentence letter** true or false (and not both).
- Also: every valuation makes every **wff** (i.e. ‘sentence’) true or false (but not both!), and we can compute the truth value of any well-formed formula (wff).

Validity in SL

Definition

An argument is **valid in SL** if there is **no** valuation in which all premises are **T** and the conclusion is **F**.

An argument is **invalid in SL** if there is **at least one** valuation in which all premises are **T** and the conclusion is **F**.

Don’t forget two *vacuous* cases of valid arguments:
(i) inconsistent premises (never all true on a valuation)
(ii) conclusion is a tautology (true on every valuation)

Disjunctive syllogism

$$\begin{array}{rl} & H \vee S \\ & {\sim}S \\ \therefore & H \end{array}$$

H	S	$(H$	$\vee$	$S)$	$\sim$	S	H	
T	**T**	**T**	**T**	**T**	**F**	**T**	**T**	✓
T	**F**	**T**	**T**	**F**	**T**	**F**	**T**	✓
F	**T**	**F**	**T**	**T**	**F**	**T**	**F**	✓
F	**F**	**F**	**F**	**F**	**T**	**F**	**F**	✓

- List all valuations for H, S.
- Compute truth values of premises, conclusion.
- Check each valuation: one premise **F**, or conclusion **T**?
- All valuations check out: valid.

An invalid argument

$H \lor S$

H

$\therefore \sim S$

H	S	$(H$	$\lor$	$S)$	H	$\sim$	S	
T	T	T	T	T	T	F	T	✗
T	F	T	T	F	T	T	F	✓
F	T	F	T	T	F	F	T	✓
F	F	F	F	F	F	T	F	✓

- List all valuations for H, S.
- Compute truth values of premises, conclusion.
- Check each valuation: one premise **F**, or conclusion **T**?
- Find a valuation with all premises **T** and conclusion **F**: invalid.

2. SL and truth tables

e. Large truth tables

Large truth tables

- For arguments with n sentence letters, there are 2^n possible valuations
  - A single letter A can be **T** or **F**: $2^1 = 2$ valuations.
  - For two letters A, B: B can be **T** or **F** for every possible valuation (2) of A: $2 \times 2 = 2^2 = 4$ valuations
  - For three letters A, B, C: C can be **T** or **F** for every possible valuation (4) of A and B: $2 \times 4 = 2^3 = 8$ valuations
  - Etc.
- In the ith reference column, alternate **T** and **F** every 2^{n-i} lines

A complex truth table

3 sentence letters A, C, E: $2^3 = 8$ lines

	A	C	E	…
1	**T**	**T**	**T**	…
2	**T**	**T**	**F**	…
3	**T**	**F**	**T**	…
4	**T**	**F**	**F**	…
5	**F**	**T**	**T**	…
6	**F**	**T**	**F**	…
7	**F**	**F**	**T**	…
8	**F**	**F**	**F**	…
	$\uparrow$	$\uparrow$	$\uparrow$	

alternate every …

4	2	1

rows

Example (simplified)

Sarah lives in Chicago or Erie.
Amir lives in Chicago unless he enjoys hiking.
If Amir lives in Chicago, Sarah doesn’t.
Amir doesn’t enjoy hiking.
∴ Sarah lives in Erie.

$$C \lor E$$

$$A \lor M$$

$$A \supset {\sim}C$$

$${\sim}M$$

$$\therefore E$$

A	C	E	M	C	$\vee$	E	A	$\vee$	M	A	$\supset$	$\sim$	C	$\sim$	M	E
T	T	T	T	T	T	T	T	T	T	T	F	F	T	F	T	T
T	T	T	F	T	T	T	T	T	F	T	F	F	T	T	F	T
T	T	F	T	T	T	F	T	T	T	T	F	F	T	F	T	F
T	T	F	F	T	T	F	T	T	F	T	F	F	T	T	F	F
T	F	T	T	F	T	T	T	T	T	T	T	T	F	F	T	T
T	F	T	F	F	T	T	T	T	F	T	T	T	F	T	F	T
T	F	F	T	F	F	F	T	T	T	T	T	T	F	F	T	F
T	F	F	F	F	F	F	T	T	F	T	T	T	F	T	F	F
F	T	T	T	T	T	T	F	T	T	F	T	F	T	F	T	T
F	T	T	F	T	T	T	F	F	F	F	T	F	T	T	F	T
F	T	F	T	T	T	F	F	T	T	F	T	F	T	F	T	F
F	T	F	F	T	T	F	F	F	F	F	T	F	T	T	F	F
F	F	T	T	F	T	T	F	T	T	F	T	T	F	F	T	T
F	F	T	F	F	T	T	F	F	F	F	T	T	F	T	F	T
F	F	F	T	F	F	F	F	T	T	F	T	T	F	F	T	F
F	F	F	F	F	F	F	F	F	F	F	T	T	F	T	F	F

Every valuation makes at least one premise false, or makes the conclusion true: the argument is valid.

2. SL and truth tables

f. Entailment, equivalence, tautologies

Validity of arguments

Definition

An argument is **valid in SL** iff every truth-value assignment either makes one or more of the premises false or it makes the conclusion true.
(i.e. there is no TVA where the premises are true but the conclusion is false).

An argument is **invalid in SL** iff at least one TVA makes all the premises true and it makes the conclusion false.

Entailment

Definition

Sentences $\mathcal{A}_1, \ldots, \mathcal{A}_n$ **entail** a sentence $\mathcal{B}$ iff every TVA either makes at least one of $\mathcal{A}_1, \ldots, \mathcal{A}_n$ false or makes $\mathcal{B}$ true.

(i.e. for any valuation where the premises are all true, the conclusion is true)

In that case we write $\mathcal{A}_1, \ldots, \mathcal{A}_n \vDash \mathcal{B}$.

The symbol '$\vDash$' is called a 'double turnstile'

Note the following relationship between entailment and validity:

$\mathcal{A}_1, \ldots, \mathcal{A}_n \vDash \mathcal{B}$ iff the argument '$\mathcal{A}_1, \ldots, \mathcal{A}_n \therefore \mathcal{B}$' is valid.

Entailment

Does $\sim(\sim A \vee \sim B), A \supset \sim C \vDash A \supset (B \supset C)$?

Note that we are asking whether two sentences of SL entail a third (i.e. do the first two provide a deductively valid argument for the conclusion $A \supset (B \supset C)$?)

Entailment

A	B	C	$\sim$	$(\sim$	A	$\vee$	$\sim$	$B)$	A	$\supset$	$\sim$	C	A	$\supset$	$(B$	$\supset$	$C)$	
T	T	T	T	F	T	F	F	T	T	F	F	T	T	T	T	T	T	
T	T	F	T T	F	T	F	F	T	T	T T	T	F	T	F F	T	F	F	$\leftarrow$
T	F	T	F	F	T	T	T	F	T	F	F	T	T	T	F	T	T	
T	F	F	F	F	T	T	T	F	T	T	T	F	T	T	F	T	F	
F	T	T	F	T	F	T	F	T	F	T	F	T	F	T	T	T	T	
F	T	F	F	T	F	T	F	T	F	T	T	F	F	T	T	F	F	
F	F	T	F	T	F	T	T	F	F	T	F	T	F	T	F	T	T	
F	F	F	F	T	F	T	T	F	F	T	T	F	F	T	F	T	F	

Tautologies

Definition

A sentence $\mathcal{A}$ is a **tautology** iff it is true on every valuation.

P	P	$\supset$	P
T	T	T	T
F	F	T	F

Contradictions

Definition

A sentence $\mathcal{A}$ is a **contradiction** iff it is false on every valuation.

P	P	$\&$	$\sim$	P
T	**T**	**F**	**F**	**T**
F	**F**	**F**	**T**	**F**

Logically equivalent sentences

Definition

Two sentences $\mathcal{A}$ and $\mathcal{B}$ are **equivalent in SL** iff every TVA either makes both $\mathcal{A}$ and $\mathcal{B}$ true or it makes both $\mathcal{A}$ and $\mathcal{B}$ false.

In other words: $\mathcal{A}$ and $\mathcal{B}$ agree in truth value, for every valuation.

Interesting case of equivalence: $\mathcal{A}$ and $\mathcal{B}$ comprise the same atomic sentence letters

Uninteresting cases: (1) all tautologies are equivalent;
(2) all contradictions are equivalent

Philosophy question: What might we take this to indicate about *the meaning* of tautologies and contradictions?

Equivalent sentences

A	B	$\sim$	A	$\vee$	B	A	$\supset$	B
T	T	F	T	T	T	T	T	T
T	F	F	T	F	F	T	F	F
F	T	T	F	T	T	F	T	T
F	F	T	F	T	F	F	T	F

Note how $(\sim A \vee B)$ wears the truth-conditions of $A \supset B$ "on its sleeves"

Equivalence and entailment

Fact

If $\mathcal{A}$ and $\mathcal{B}$ are equivalent, then $\mathcal{A} \vDash \mathcal{B}$ (likewise, $\mathcal{B} \vDash \mathcal{A}$).

Proof

- Look at any valuation: it makes $\mathcal{A}$ true or false.
- If $\mathcal{A}$ is false, the valuation is not a counterexample.
- If $\mathcal{A}$ is true, $\mathcal{B}$ is also true (since $\mathcal{A}$ and $\mathcal{B}$ agree in truth value on every valuation).
- So if $\mathcal{A}$ is true, the valuation is also not a counterexample.
- So, no valuation can be a counterexample to $\mathcal{A} \vDash \mathcal{B}$.

Two Questions about Entailment

Let '$\Gamma \nvDash \Psi$' mean that the (set of) sentence(s) Γ does not semantically entail Ψ, i.e. an argument from Γ to Ψ is invalid

1. True or False? If $\Phi \vDash \Psi$, then $\sim\Phi \nvDash \Psi$

2. True or False? If $\Gamma \vDash \Phi$ and $\Delta, \Phi \vDash \Psi$, then $\Gamma, \Delta \vDash \Psi$?

2. SL and truth tables

g. Consistency

Consistency (a.k.a joint satisfiability)

Definition

Sentences $\mathcal{A}_1, \ldots, \mathcal{A}_n$ are **consistent** (i.e. ‘satisfiable’) in SL if there is at least one TVA that makes all of them true.

If they are not satisfiable, we say that they are **inconsistent** (a.k.a ‘jointly unsatisfiable’).

$A \vee B$, $\sim A$, B are consistent (a.k.a ‘satisfiable’).

$A \vee B$, $\sim A$, $\sim B$ are inconsistent (a.k.a ‘unsatisfiable’).

Inconsistency and validity

- Any argument with inconsistent premises is valid.
  - If premises are inconsistent, no valuation makes them all true.

    $\Rightarrow$ No valuation makes them all true and the conclusion false.

    $\Rightarrow$ No valuation can be a counterexample.
- An argument is valid if, and only if, the premises together with the **negation** of the conclusion are inconsistent.
- This fact is the basis our tree method for validity ('STD')!

LSAT puzzle: Can you send Amir in the boat?

A, B, C, D: Amir, Betty, Chad, Dana are in the boat.

Amir won't go without Chad. (If no Chad, then no Amir)

$$A \supset C$$

Chad only goes if at least one of Betty and Dana goes too.

$$C \supset (B \vee D)$$

Amir and Dana can't be in the boat together.

$$\sim(A \,\&\, D)$$

$$A \supset \sim D$$

$$\sim A \vee \sim D$$

Dependency resolution by SAT checking

A, B, C, D: package A, B, C, D is installed.

Package A depends on package C.

$$A \supset C$$

Package C requires either package B or D.

$$C \supset (B \vee D)$$

Package A is incompatible with package D.

$$\sim(A \mathbin{\&} D)$$

$$A \supset \sim D$$

$$\sim A \vee \sim D$$

Solution as satisfiability question

Can you send **Amir** in the boat?

Can package **A** be installed?

Same as: Are these sentences consistent?

$$\mathbf{A}$$

$$A \supset C$$

$$C \supset (B \vee D)$$

$$\sim(A \,\&\, D)$$

More complex satisfiability questions

Can you send **Amir without Betty** in the boat?

Can package **A** be installed **without installing B**?

Same as: Are these sentences consistent?

$$A \mathbin{\&} {\sim}B$$

$$A \supset C$$

$$C \supset (B \vee D)$$

$${\sim}(A \mathbin{\&} D)$$

(Exercise: construct a complete truth table. Which valuations, if any, satisfy all four sentences?)

Complexity of logical testing

- In general, testing for validity, satisfiability, tautology, etc., requires making a complete truth table
  - Testing for validity requires checking **every** valuation (although can halt if find a counterexample to validity).
  - Testing for satisfiability requires finding **at least one** valuation.
- If there are n sentence letters, there are 2^n valuations to check.
- Computer scientists have yet to find a method that can (always) do this faster than truth tables (connected to “P vs NP problem”).
- See Cook-Levin Theorem

More on Complexity Classes

- **Class P**: solvable in polynomial time by a deterministic Turing machine (DTM)
- **Class NP**: solvable in polynomial time by a non-deterministic Turing machine (NTM). Equivalently: can *verify/check* solutions by DTM in polynomial time
- Boolean satisfiability and validity problems are *NP-complete*:
  - Problems that are easy to check, but difficult to solve
  - If we could solve these in polynomial time with DTM, then we could solve *ANY* NP problem in polynomial time
  - You might stand to make a lot of money (you would potentially have an efficient algorithm for many problems)!!!
  - (At the very least, you could cash in for a million-dollar prize)
  - But most experts think that class P $\neq$ class NP

3. Mathematical Induction & Recursive Definitions

3. Mathematical Induction & Recursive Definitions

a. Intro to Math. Induction

A Super-natural example!

- You're doing time in purgatory, and are automatically enrolled in philosophy (lucky you!)
- Good news: To pass your class, you just need to produce one satisfactory piece of work
- Bad news bears: your teacher is Gordon Ramsay, and he is never satisfied with any assignment
- Whenever you turn something in, Ramsay returns it with suggestions for improvement, with revisions due tomorrow
- How long should you expect to spend in this class?

Mathematical Induction for SL Sentences

If you want to prove that **ALL** sentences (wffs) of SL have a particular property, it suffices to show the following:

1. All **atomic** sentences have that property ('**base case**')
2. If $\mathcal{A}$ and $\mathcal{B}$ are two **arbitrary wffs** with that property, then so are the sentences built out of them by adding a connective

   '**Induction step**' (There are five cases to consider:)

   (i) $\sim\mathcal{A}$

   (ii) $(\mathcal{A} \,\&\, \mathcal{B})$

   (iii) $(\mathcal{A} \vee \mathcal{B})$

   (iv) $(\mathcal{A} \supset \mathcal{B})$

   (v) $(\mathcal{A} \equiv \mathcal{B})$

A Simple Example

- Prove by induction that every well-formed formulae (wff) of SL has exactly as many left parentheses as it has right parentheses

- Don’t forget to explicitly state the **base case**

- Don’t forget to explicitly state the **Induction Step**!

Three More Examples!

Prove the following by induction. Don't forget to explicitly state the base case and the induction step!

1. No wff begins or ends with a binary connective
2. No wff contains two consecutive binary connectives (i.e. with no symbols between them)
3. If a wff doesn't contain any binary connectives, then it is contingent. (hint: say that a wff is *baller* if it either contains a binary connective or is contingent. Use induction to show that every wff is baller.)

Two DISTINCT notions of ‘Induction’

- **Mathematical Induction** is completely different from **scientific induction** or what we earlier called ‘inductive arguments’ (which we contrasted with ‘deductive arguments’)
- **Mathematical Induction**: Rigorous method of proving that a property holds for an infinite number of cases. Follow the steps we’ve outlined!
- **Scientific Induction**: fallible method of supporting a claim, based on a finite number of previous observations.
  - All of the ravens I have ever seen are black
  - Therefore, the next raven I see will be black
  - Notice how your conclusion could be wrong! Albino ravens!

3. Mathematical Induction & Recursive Definitions

b. Recursive Definitions

A Non-recursive Motivation for Recursion

- When a bigger thing can be studied in smaller chunks, it is often fruitful to do so
- Descartes' *Rules for the Direction of the Mind*:
  break problems down to their simplest parts and have a secure grasp of them before proceeding
- Or as we might say: Show me complexity, and I will find simplicity (Gotta get this on a T-Shirt!)
- Recursive definitions are one instance of applying this rule

A Non-recursive approach to Language Learning

- Some tourist manuals will have lists of sentences to phonetically memorize before travelling: learn how to say “Where is the train to the airport?” or “Where can one find a drugstore?” etc.
- The plan is to commit all these to memory, by brute force.
- You will know what “Hvor er toget til lufthavnen”, “Hvor finder man et apotek?” etc. mean because you have seen them before and can recall them.
- But this is entirely uncreative. You can only say/understand things you have seen before using this technique.

Recursion as the basis of English

- But much of the time, we say things that we've *never* said or heard before, yet we understand what they mean.
  - "Your fate awaits in the office of the chair of the Corporation"
  - I would bet no one in class has heard this sentence before.
  - But (maybe) you understand what it means.
- What makes this possible is that our language has a recursive structure – we have basic components and then rules for making more complex, meaningful expressions out of them.

Example: Possessive of a noun phrase

- We can iterate grammatical rules, i.e. apply them again and again.
- For example we have a way to form an expression “[Noun Phrase]’s son”.
- I can apply this to the name ‘Andrea’ to get “Andrea’s son”.
- Then I can apply it again, to get “Andrea’s son’s son”.
- And again, to get “Andrea’s son’s son’s son”.

Example: nested object of verb phrases

- Sometimes, sentences can be so complicated that we need to break them down into components and study their structure to figure out what they mean
- But if we understand the words, and we understand the rules of grammar, we can figure out the recursive structure
- For example, the following is a meaningful sentence of English:
- *The rat the cat the dog bit chased ate the cheese*
- What is this rat smoking??? It’s not obvious that this is a meaningful sentence, never mind what it means!

Break it down!

- To figure out what it's saying, let's take a simpler part. Say that you see a rat eating some cheese.
- You could say: "The rat ate the cheese." Nothing obscure here
- But maybe there were several rats running around, and only one of them ate the cheese.
- You want to indicate which one, and it happens to be the one that the cat chased.
- So you can form the noun phrase "The rat the cat chased", and state "The rat the cat chased ate the cheese."
- The rat $\underbrace{\Longrightarrow\Longrightarrow}_{\text{Which rat?}}$ The rat the cat chased

Still breaking...

- And perhaps there are lots of cats running around too.
- You want to say specifically that you are referring to the cat that the dog bit.

  The cat $\underbrace{\Longrightarrow\Longrightarrow}_{\text{Which cat?}}$ The cat the dog bit.

  So take the phrase “The rat the cat chased”, and replace “the cat” with “the cat the dog bit” to get the noun phrase:
- “The rat the cat the dog bit chased”:
- And you can state that this particular rat, chased by that particular cat [the cat the dog bit] ate the cheese:
- “The rat the cat the dog bit chased ate the cheese.”

More Examples of our Beautiful Recursive Language

- Now that you know the rules, and can apply them over and over, you can figure out what this means.
- “The rat the cat the dog the farmer bought bit chased ate the cheese.”
- Some others to try for yourself (not essential to the course, for entertainment value only:)
- Fish fish fish fish fish.
  Fish fish fish fish fish fish fish

A Moral of our Convoluted Story

- **The point**: if you know the basic units (in this case: the words; in *SL* the sentence letters) and you know the rules for creating more complex meaningful parts, you open up an astonishing range of possibilities.
- Definition by Recursion is the canonical way to define objects or structures that depend on iterated rules applied to a given basis
- Proof by Induction is a powerful way to prove things about structures that are defined in this way.

Recursive Definition of Sentences of SL

Recall that we define the well-formed formulae (wffs) *recursively*:

1. **Base Clause**: Each atomic sentence is a wff
2. **Recursion Clause(s)**:
    - If $\mathcal{P}$ is a wff, then so is $\sim\mathcal{P}$
    - If $\mathcal{P}$ and $\mathcal{Q}$ are both wffs, then so are:
        - $(\mathcal{P} \,\&\, \mathcal{Q})$
        - $(\mathcal{P} \vee \mathcal{Q})$
        - $(\mathcal{P} \supset \mathcal{Q})$
        - $(\mathcal{P} \equiv \mathcal{Q})$
3. **Closure clause**: Nothing else is a wff of SL

Natural Numbers

- The set of natural numbers $\mathbb{N} = \{0, 1, 2, 3, 4, 5, \ldots\}$, beginning with 0, is the paradigm of a set of objects with a recursive definition.
- The first natural number is 0, and larger numbers are generated by adding 1.
- Any whole number greater than 0 can be reached by starting with 0 and iterating this operation.
- (N.B.: Sometimes $\mathbb{N}$ is defined excluding '0';
  but remember: we are inclusive!!!)

Recursive Definition of “natural number”

- Defining the set of natural numbers $\mathbb{N} = \{0, 1, 2, 3, 4, 5, \ldots\}$:
- Put the definition into our fave recursion template:
  - **Base clause**: 0 is a natural number.
  - **Recursion clause**: If n is a natural number then
    $n + 1$ is a natural number.
  - **Closure clause**: Nothing else is a natural number.
- This definition generates the entire set $\mathbb{N} = \{0, 1, 2, 3, 4, 5, \ldots\}$, starting with 0 and repeating the operation of “$+1$”.

Strings! (of letters—not of physics!)

- It can be useful to treat the set of *all strings of letters* in a given alphabet recursively
- Say we take a fixed alphabet with just the characters $\{a, b\}$.
- The basic strings will be the single characters a and b.
- More complex objects are built up by the operation that inputs a string x and a member u of the alphabet, and that outputs the string 'xu' that results from appending 'u' to the right of 'x'
- Any string can be built up from a single letter by iterating this operation—in effect, simply spelling out 'x' left-to-right.

An Example: Recursive Definition of Strings

- Let the alphabet be $\{a, b\}$ and consider the string ‘bba’: we obtain this string as follows:
    - Start with ‘b’. Append ‘b’ to ‘b’ and obtain ‘bb’
    - Finally, append ‘a’ to ‘bb’ and obtain ‘bba’
- Since we can generate every string this way, we can give a recursive definition of “string of letters from the alphabet $\{a, b\}$”:
    1. **Base clause**: a and b are strings from $\{a, b\}$
    2. **Recursion clause**: If x is a string from $\{a, b\}$ then xa and xb are strings from $\{a, b\}$
    3. **Closure clause**: Nothing else is a string from $\{a, b\}$

Concatenation of the Feline Nation

- A note on terminology: when you stick together two strings of symbols, we say that you “concatenate” them

- Sometimes, the symbol ‘$*$’ is used to indicate concatenation, so “$x * y$” means “the result of concatenating the string x and the string y”.

- Example: ‘$aaba * cca$’ is just the string ‘aabacca’

Recursively Defining ‘Proofs’

- When we discuss metalogic, we will use the fact that *proofs* provide yet another domain that can be given a recursive definition
- So the set of proofs is another domain where inductive arguments can be given
- A generic description for now:
- Proofs are constructed by applying rules of inference to premises
- The simplest proofs are axioms or single premises:
  - Any axiom is itself a one–step proof of itself, a single premise is a one–step proof of itself, from itself.

Recursively Defining ‘Proofs’

- More complex proofs are built up from simpler ones by adding a step, which must either be an axiom or premise, or follow by a rule from earlier lines in the proof
- That is: something is a proof if it is an axiom or premise, or it is obtained from a proof by applying one of a finite number of rules!
- So it’s a recursive structure: A basis, iterated construction procedure, and nothing else:
  1. **Base clause**: An axiom or a premise is a proof
  2. **Recursion clause**: If **Pr** is a proof, then the result of applying one of the rules of our system [more on those later!] to make **Pr** one line longer, is a proof
  3. **Closure clause**: Nothing else is a proof

3. Mathematical Induction & Recursive Definitions

c. Mathematical Induction

Mathematical Induction: Motivation

- One reason to pay attention to definition by recursion: it makes possible the proof technique of "(mathematical) induction".
- Don't confuse this use of 'induction' with "(scientific) induction", i.e. drawing probable inferences from observations (vs. "deduction", which draws certain inferences by reasoning alone)
- We have inductive evidence that you'll be doing a lot of induction!
- Mathematical induction can be used in any domain where some objects can be singled out as basic, and where complex objects are built up out of simpler ones by iterating an operation.

General Pattern for mathematical induction for $\mathbb{N}$

- If you can prove two facts (called the **Base case** and the **Induction step**) about a property C, then you know it holds for every natural number:
    - **Base case**: C holds of 0. (or 1, or whatever you are starting the sequence with.)
    - **Induction Step**: Assume that C holds of some given number n. (This assumption is called the *Induction Hypothesis*). Using this assumption, prove that C holds of $n + 1$.
- ALWAYS REMEMBER TO EXPLICITLY NOTE BOTH THE **BASE CASE(s)** AND THE **INDUCTION STEP**!

A Simple Example of Induction over $\mathbb{N}$

- Prove that the sum of the first n natural numbers is $\frac{1}{2}n(n+1)$ for any natural number n
  - (start at 1 and ignore 0, because it contributes nothing to the sum)
- Some notation: to write a sum of a list of numbers, you need to index them to $1, 2, \ldots n$ to get a list $m_1, m_2, \ldots m_n$. Write this as:
- $\displaystyle\sum_{i=1}^{n} m_i$ (using a capital greek 'Sigma').
- Now it is particularly easy to write the sum of the first n natural numbers: you don't need to index them to anything since they are in order already!
- So just write: $\displaystyle\sum_{i=1}^{n} i = 1 + 2 + 3 + \ldots + n$

Simple Example: checking some cases

- Often helpful to orient yourself by checking some cases:
- Note that:
  $1 = \frac{1}{2}(1)(1+1) = \frac{2}{2} = 1$
- $1 + 2 = \frac{1}{2}(2)(2+1) = 3$
- $1 + 2 + 3 = \frac{1}{2}(3)(3+1) = (3)(2) = 6$
- $1 + 2 + 3 + 4 = \frac{1}{2}(4)(4+1) = (2)(5) = 10$
  Each of these is an instance of the formula $\displaystyle\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$
- When you try the formula in simple cases, it works. You might conjecture that it holds generally – but that isn't a *proof*.

Simple Example: Motivating proof by Induction

- How can we prove that this formula holds generally?
- Note this foothold: there is a way to break down each case so that it depends straightforwardly on the immediately previous one.
- Say that I ask you to compute $1 + 2 + 3 + 4 + 5$ and you already know that $1 + 2 + 3 + 4 = 10$
- Note that $1 + 2 + 3 + 4 + 5 = \underbrace{[1 + 2 + 3 + 4]}_{\text{instance of prior case}} + 5$
- Simplifies to $10 + 5 = 15$, given what you already know.

Simple Example: Motivating proof by Induction

- Of course, "$1 + 2 + 3 + 4 + 5$" is a small enough number that it's easy to just add the numbers together without bothering with simplifying tricks.
- But with bigger numbers that becomes less and less of an attractive option. Using the simplifying trick saves loads of time.
- It might take some time to add up the first 1,000,000 numbers, but if I already know the sum of the first 999,999, my job is easy:
- Just take *that* sum and add 1,000,000 to it!

An Inductive Proof of this Fact

- Now we want to **prove** the formula for the sum of the first n numbers:
- $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$
- So we first need to prove the base case. In this problem, the base case is $n = 1$.
- So we've actually already covered the base case:
- **Base Case:** $1 = \frac{1}{2}(1)(1+1) = \frac{2}{2} = 1$

A Concern about the Induction Hypothesis

- For the Induction Step, we first **assume** that we already know, *for some given k*, that $\sum_{i=1}^{k} i = \frac{1}{2}k(k+1)$
- We call this assumption "*The Induction Hypothesis* (IH)".
- At this point there may be howls from the bleachers: Isn't this just assuming what you have to prove??????
- Good question! It shows you're paying attention. But no: it isn't assuming what you have to prove
- In the IH, we could have even used 'n' rather than 'k'!

Induction Hypothesis

- What we want to prove is that the formula is true for every $n \in \mathbb{N}$. The induction hypothesis assumes we know the formula is true for some **specific**, undetermined value k.
- We will then **use** the hypothesis to prove the case for $k + 1$.
- We then **stop** assuming the induction hypothesis, and instead conclude a conditional claim:
- **If** the induction hypothesis is true for a number k, **then** the relevant property is also true for $k + 1$

Induction Hypothesis

- Given the induction hypothesis, $\sum_{i=1}^{k} i = \frac{1}{2}k(k+1)$ we want to prove the formula holds for $k+1$, i.e. that:
- $$\sum_{i=1}^{k+1} i = \frac{1}{2}(k+1)(k+2)$$

  [**Note:** We assume the formula is true for some number k, and we prove, *given this assumption*, that it must be true for $k+1$ as well.]

Carrying out the Induction Step

- Our target is to show that $\sum_{i=1}^{k+1} i = \frac{1}{2}(k+1)(k+2)$
- OK, let's remember what we noted above, that:

$$1 + 2 + \ldots + (k+1) = \underbrace{[1 + 2 + 3 + \ldots + k]}_{\text{instance of prior case}} + (k+1)$$

- So we can simplify: $1 + 2 + 3 + \ldots + (k+1) = [\frac{1}{2}k(k+1)] + (k+1)$
- Some algebra: $[\frac{1}{2}k(k+1)] + (k+1) = [\frac{1}{2}(k^2+k)] + [\frac{2k}{2} + \frac{2}{2}]$
$= \frac{k^2+3k+2}{2} = \frac{(k+1)(k+2)}{2} = \frac{1}{2}(k+1)(k+2)$
- And that's what we wanted to prove!

Recapping the Structure of Our Inductive Proof

- Recapping our Proof Steps (remember these steps!)
  1. **Base Case**: showed that $\sum_{i=1}^{n} i = \frac{1}{2}n(n+1)$ is true when $n = 1$
  2. **Induction Step**: showed that for any k, **if** $\sum_{i=1}^{n} i = \frac{1}{2}n(n+1)$ is true when $n = k$ **then** $\sum_{i=1}^{n} i = \frac{1}{2}n(n+1)$ is true when $n = k + 1$.
- These two facts entail that $\sum_{i=1}^{n} i = \frac{1}{2}n(n+1)$ when $n =$ **any** $k \in \mathbb{N}$, i.e. for all $n \in \mathbb{N}$

Why does Mathematical Induction Work?

- It's easy to see how this works: say we have a specific number.
  - To be concrete, say that I give you 325.
  - How do you know that the formula holds when $n = 325$?
- Well, the formula holds for $n = 1$ (base case) and so by applying the induction step to the base case you know that the formula holds for $n = 2$. Applying the induction step to $n = 2$ tells you the equation holds for $n = 3$ . . .
- Applying the induction step 324 times in this fashion gives you that the equation holds for $n = 325$.
- BOOM!
  - Feel that? That's the power of induction.

A Non-inductive Proof of the Same Fact

- Here is a quick proof that, according to legend, the great mathematician Gauss found by himself when he was 7 years old
- Take the sum $\sum_{i=1}^{n} i = 1 + 2 + 3 + \ldots + n$ and write it twice, one above the other, the second one in the "most to least" order, and add the series term by term. That is,
- $$\begin{array}{l} \sum_{i=1}^{n} i = 1 + 2 + 3 + \ldots + n \\ \sum_{i=1}^{n} i = n + (n-1) + (n-2) + \ldots + 1 \\ \hline 2 \times \sum_{i=1}^{n} i = \underbrace{(n+1) + (n+1) + \ldots (n+1)}_{n \text{ terms}} \end{array}$$

Non-inductive Proof Continued

- $2 \times \sum_{i=1}^{n} i = \underbrace{(n+1) + (n+1) + \ldots (n+1)}_{n \text{ terms}}$
- So we get $2 \times \sum_{i=1}^{n} i = n(n+1)$ and hence $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$
- Which is what we were looking for, proved without induction
- But notice how—in contrast to kid Gauss—induction doesn't require any special insight or inspiration!

An Example to Work Through in Class!

- An analogue of a HW Problem:
- Prove that for all $n > 3$, $n \in \mathbb{N}$, $2^n < n!$

  where $n! := \underbrace{(n \times (n-1) \times \ldots 3 \times 2 \times 1)}_{n \text{ times}}$

- Just carry out the steps, noting each explicitly:

  1) What is the **Base Case** we need to show?

  2) What is our **Induction step**?

     i) Induction hypothesis?; ii) What do we need to show?

  3) Just do it! (Note what you've proven)

An Example to Skip in Lecture!

- Another numerical example with the same flavor (i.e. *tasty*!)
- $\sum_{i=1}^{n} i^2 = \frac{1}{6} n(n+1)(2n+1)$
- The general strategy for attacking this problem inductively:

  1) (**Base Case**) prove $\sum_{i=1}^{1} i^2 = \frac{1}{6}(1)(1+1)(2(1)+1)$

- Just elementary arithmetic (typically base cases are *chill*)

Another example continued

2) (**Induction Step**): assume that for a given k,

$$\sum_{i=1}^{k} i^2 = \frac{1}{6}k(k+1)(2k+1)$$

- Then set out to show that

$$\sum_{i=1}^{k+1} i^2 = \frac{1}{6}(k+1)(k+2)(2(k+1)+1)$$

- The trick is to find a way to *use* the information you are given in the induction hypothesis
  - Break down the $n = k+1$ case so it consists of the $n = k$ case plus some comparatively simple other stuff.
  - This is usually the step that requires the most thought

Working through the Deets

- Happily, when we are dealing with sums, it is easy to simplify the $n = k + 1$ case by appealing to the $n = k$ case:

$$\sum_{i=1}^{k+1} i^2 = \underbrace{1 + 2 + 3 + \ldots k}_{\text{This is } \sum_{i=1}^{k} i^2} + (k+1)^2$$

$$= [\textstyle\sum_{i=1}^{k} i^2] + (k+1)^2$$

$$= \underbrace{\frac{1}{6}k(k+1)(2k+1)}_{\text{Substituting } \frac{1}{6}k(k+1)(2k+1) \text{ for } \sum_{i=1}^{k+1} i^2} + (k+1)^2.$$

Now we have an equation that doesn’t have the sum operator in it at all.

Working through more Deets

$$= \underbrace{\frac{1}{6}k(k+1)(2k+1)}_{\text{Substituting } \frac{1}{6}k(k+1)(2k+1) \text{ for } \sum_{i=1}^{k+1} i^2} + (k+1)^2.$$

- We now have a *much* simpler problem in front of us:
- Consider whether we can algebraically reduce $\frac{1}{6}k(k+1)(2k+1) + (k+1)^2$ to

  $\frac{1}{6}(k+1)(k+2)(2(k+1)+1)$.
- Spoiler alert!
- It is possible. (But I would not pay $$$ for this show)

3. Mathematical Induction & Recursive Definitions

d. Ordinary vs. Complete Induction

Ordinary Induction vs. Complete Induction

- Proofs by mathematical induction fit into two slightly different schemes, depending on how much you assume in the induction hypothesis about the cases that are less complex than the one you set out to prove.
- These argument patterns are equally rigorous, but they differ a bit in their logical structure, so it is worth pointing out the difference in form explicitly.
- In the cases we have just considered, we first prove some property is true of 0 (or 1, or whatever else is the first(s) in the series)
- Second, we prove that if we assume that property is true of an arbitrary n, we can prove it holds for the $(n+1)$–th case

Complete induction (a.k.a. ‘strong induction’)

- The variation is sometimes called the method of **complete induction** (or ‘**strong induction**’).
- In complete induction, we modify the induction hypothesis:
  1. We prove the base case as before
  2. We assume that the property holds of **every** number less than a given k (adding the assumption that k is greater than whatever number was considered in the base case)
- The difference is that instead of assuming the thesis *just* for the number preceding a given one, we assume it for **every** number less than the given one
- Each method is equally rigorous: it just happens that one form is convenient for some problems, the other for others

Ordinary (a.k.a Weak) vs. Complete (a.k.a Strong) Induction

Ordinary mathematical induction:

Base Case: $C(x)$ holds of the stuff in base clause, e.g. `0`

Induction Step: Take any case n. If $C(x)$ holds of case n, then $C(x)$ holds of $n+1$

Complete induction:

Base Case: $C(x)$ holds of the stuff in base clause, e.g. 'a'

Induction Step: Take any $k >$ `0` [Or $k >$ the base case index(s)]. If $C(x)$ holds for **every** $x < k$, then $C(x)$ holds of k.

- In both cases, if you can prove the Base Case and the Induction Step, you can conclude that $C(x)$ is true for every x (yee haw!)

A time to be Complete. A time to be Strong

- In particular: When dealing with strings of symbols in a language, the method of complete induction tends to be more useful.
- Illustration: you are dealing with sentences in *SL*, doing induction on the number of connectives in the sentence
- Say that $\mathcal{S}$ is a sentence with $n > 0$ connectives, and $\mathcal{S} = \mathcal{S}_1 \vee \mathcal{S}_2$.
- Then you know that both $\mathcal{S}_1$ and $\mathcal{S}_2$ have fewer connectives than $\mathcal{S}$, but you don't know precisely how many either of them has.
- The n connectives in $\mathcal{S}$ could be divided up in lots of different ways between $\mathcal{S}_1$ and $\mathcal{S}_2$, so the hypothesis used in Complete Induction is more useful: it covers all the possibilities
- Language *SL* says “stay strong!”

3. Mathematical Induction & Recursive Definitions

e. Recursion and Induction for Palindromes

Little Languages!

- One of the problems on the problem set is aimed at helping you get accustomed to doing induction/recursion on language structures
- In this case the "language" is pretty rudimentary: the set of all strings you can make by concatenating the characters $\{a, b\}$
- Let's spice things up by extending this alphabet to include the great big beautiful 'c': $\{a, b, c\}$.
- This is useful for providing a simple example of complete induction

Palindromes semordnilaP

- A **palindrome** is a string of letters that reads the same way backwards and forwards (ignoring spaces and punctuation). One simple example is “race car”.
- In case that example went by too fast: “Yo, banana boy!” “Do geese see God?”
- There are some trivial cases too: “I” is an English language palindrome, and so is “a”. Aha!
- *Let us turn now* to our rudimentary language of the set of all strings in the alphabet $\{a, b, c\}$, to study palindromes there (We sometimes wonder: what if the reader said at this juncture, “No! Thou shall not turn now!” ?)

sthgisnI peeD (Deep Insights!)

- Say we want to prove things about palindromes (and boy, do we ever!)
- Here is an insight that makes induction an option:
- If you remove the first and last letter of a palindrome, you get *another* palindrome, two letters shorter

  ($\Rightarrow$ *insert mind-blown emoji here*$\Leftarrow$)
- For instance: "*aabbaabbaa*" is a palindrome. If you remove the first and last '*a*' then you get "*abbaabba*".
  That's a palindrome too
- And so is "*bbaabb*", etc, tracing right back to "*aa*"
  (tracing back even further, to the empty string 'ϵ', if we're string half-empty kinda people. 'What's an empty string?' you might say: oh, it's nothing)

Recursively Defining Palindromes

- This mind-blowing fact lets us recursively define "palindrome"
- We can use the structure we've discovered to incorporate useful info about the set of palindromes into a recursive definition.
- Call a string over $\{a, b, c\}$ a **recursive palindrome** if it satisfies the three conditions:
  1. **Base clause**: 'a', 'b', 'c', 'aa', 'bb', 'cc' are recursive palindromes
  2. **Recursion clause**: If s is a recursive palindrome, then $a * s * a$, $b * s * b$, and $c * s * c$ are recursive palindromes
  3. **Closure clause**: Nothing else is a recursive palindrome in $\{a, b, c\}$

Induction over Palindromes

- What we want to do now is **prove** that every palindrome in $\{a, b, c\}$ is a recursive palindrome, i.e. that our recursive definition captures the intuitive phenomena
- We'll do induction on *the length of a string s*
- Note that what follows is NOT what you're being asked to do on PS 3, problem 1(ii).
- 1(ii) asks you to prove a specific property of a–palindromes

Base Case of our Induction

- **Base case:** Say that s is 1 or 2 symbols long. Then:

  i) it is one of 'a', 'b', 'c', 'aa', 'bb', or 'cc', in which case it is both a recursive palindrome and a palindrome

  ii) It is one of 'ab', 'ba', 'bc', 'cb', 'ac', or 'ca' in which case it is neither a palindrome nor a recursive palindrome
- (Argument for ii): None of the strings listed in ii) can be an R–palindrome because any R–palindrome that isn't called an R–palindrome by the base case must be called an R–palindrome by the recursion clause.

  -But if s is an R–palindrome by the recursion clause, then it must contain at least three letters.)

Stating the Induction Step

- **Induction step**: *Assume* (Induction hypothesis) that every palindrome of length less than k symbols is a recursive palindrome and conversely, where $k > 2$
- Note that the “and conversely” means we need to prove “both directions”
- You WON’T need to prove both directions on your problem set!

Need to show: s a palindrome $\Rightarrow$ s a recursive palindrome

- Let s be an arbitrary palindrome with exactly k symbols, $k > 2$
- Since s has three or more symbols, we can remove the first and last letter, leaving a string s'.
- Since s reads the same backwards and forwards, and s' comes from s by removing the first and last letter, s' reads the same backwards as forwards. That is, s' is a palindrome.
- Since s' is a palindrome, and it has fewer than k letters, then it is an R–palindrome by the induction hypothesis
- Since s is a palindrome, the first and last letter must be the same, i.e. either 'a', 'b', or 'c'. So $s = a * s' * a$, or $s = b * s' * b$, or $s = c * s' * c$, where as we have shown s' is an R–palindrome.
- So, by the recursion clause of the definition of R–palindrome, s is an R–palindrome.

NTS: s a palindrome $\Leftarrow$ s a Recursive palindrome

- Let s be an arbitrary R–palindrome with exactly k symbols, $k > 2$
- Since $k > 2$, s cannot be an R–palindrome due to the base clause, so it must be counted by the recursion clause
- Hence $s = a * s' * a$, or $s = b * s' * b$, or $s = c * s' * c$, where s' is an R–palindrome, by the recursion clause of the definition.
- Since s' has fewer than k symbols, it falls in the scope of the induction hypothesis, so s' is a palindrome. Since s' is a palindrome, therefore $s = a * s' * a$, or $s = b * s' * b$, and $s = c * s' * c$ are palindromes because they will also read the same backwards and forwards.
- Since one of these is s, s is a palindrome
- This completes the inductive proof. And so we ask ourselves: Are we not pure? “No, sir!” Panama’s moody Noriega brags. “It is garbage!” Irony dooms a man—a prisoner up to new era.

Remarks on the “a−palindrome” question on the HW

- Here’s the problem:
- Call a string over $\{a, b\}$ an “a−palindrome” if it is a palindrome that has “a” as the middle letter.
- (i) Give a recursive definition of “a − palindrome”, and (ii) prove by induction that every a−palindrome has an even number of “b”’s.

Two things to do! (each with substeps)

- There are two things you need to do (each with substeps):
  1. Give a recursive definition of “a–palindrome”. For ease of reference, say that the definition defines the “recursive a–palindromes”
  2. Using induction, show that every recursive a–palindrome has an even number of b’s
- (In principle, you would also have to prove that the two definitions pick out the same set. That is, that a–palindromes are exactly the recursive a–palindromes. But I’m NOT asking you to do this as part of PS 3)

Modifying an Earlier Analogue problem

- Now step (1) is almost identical to the treatment of what are called "recursive palindromes" above. In that more complicated case, focusing on alphabet $\{a, b\}$, we'd say:
  1. **Base clause**: 'a', 'b', 'aa', 'bb' are recursive palindromes
  2. **Recursion clause**: If s is a recursive palindrome, then $a * s * a$, and $b * s * b$ are recursive palindromes
  3. **Closure clause**: Nothing else is a recursive palindrome in $\{a, b\}$
- The definition of a–palindrome is *almost identical* to this. You really just need to modify one of the clauses
- Then use induction to prove the claim about a–palindromes having an even number of "b"s

Baby Stamps!

- Let's work through an analog of PS 3, problem 3
- Baby Stamps: prove by induction that if you have (an unlimited supply of) 2¢ and 11¢ stamps, you can make any integer amount greater than or equal to 10¢

1. Base Case(s): more complicated than usual!
2. Induction Step: hint—use COMPLETE induction, so consider a specific $k >$ largest base case. Then state Induction hypothesis as holding for all n less than k but ≥ 10

Hints for Question 4, PS 3

- Hint: the trick is to figure out *what* to do induction *over*!
- In the base case, consider two arbitrary odd numbers, and use the hint on the PS to express them (i.e. a natural number d is odd if and only if there exists another natural number k such that $d = 2k + 1$).
- Use complete induction!

3. Mathematical Induction & Recursive Definitions

f. Recasting Induction in SL as Complete Induction

Special Induction Schema for the language *SL*

- Recall that for SL, we can use a special induction schema:
- If you want to prove that **ALL** sentences (wffs) of SL have a particular property, it suffices to show the following:
  1. All **atomic** sentences have that property ('**base case**')
  2. If $\mathcal{A}$ and $\mathcal{B}$ are two **arbitrary wffs** with that property, then so are the sentences built out of them by adding a connective '**Induction step**' (There are five cases to consider:)
     - (i) $\sim\mathcal{A}$
     - (ii) $(\mathcal{A}\,\&\,\mathcal{B})$
     - (iii) $(\mathcal{A} \vee \mathcal{B})$
     - (iv) $(\mathcal{A} \supset \mathcal{B})$
     - (v) $(\mathcal{A} \equiv \mathcal{B})$

Complete Induction for the language *SL*

- Implicitly, this special schema is Complete Induction over the string length k of an SL sentence
- To simplify, let's use a fragment of *SL* that contains just atomic sentences and sentences whose only connective is "&"
- We can use induction because this rudimentary part of *SL* has a recursive definition:
  1. **Base clause**: Every atomic sentence is a sentence of *SL*
  2. **Recursion clause**: If $\mathcal{P}$ and $\mathcal{Q}$ are sentences of *SL*, then $(\mathcal{P} \,\&\, \mathcal{Q})$ is a sentence of *SL*
  3. **Closure clause**: Nothing else is a sentence of (our fragment of) *SL*

Complete Induction for the language *SL*

- Let’s reprove this claim (for our fragment): every sentence of SL has an equal number of left and right parentheses
- Proceed by (complete) induction on the number n of symbols in the sentence $\mathcal{S}$, i.e. the string-length:
- **Base Case**: $n = 1$

  Then $\mathcal{S}$ is an atomic sentence, so it has no parentheses. Thus there are the same number of right and left parentheses.

Starting the Induction step

- **Induction step**: Let $\mathcal{S}$ be an arbitrary sentence of SL with k symbols, where $k > 1$. Assume that every sentence of this language with fewer than k symbols has the same number of left as right parentheses (Induction hypothesis)
- $\mathcal{S}$ is not a sentence letter, so it must be of the form $(\mathcal{S}_1 \,\&\, \mathcal{S}_2)$, with $\mathcal{S}_1$ and $\mathcal{S}_2$ sentences of *SL* (since we're focusing on the conjunctive fragment of SL)
- Note that $\mathcal{S}_1$ and $\mathcal{S}_2$ have fewer than k symbols
- So by the induction hypothesis, both $\mathcal{S}_1$ and $\mathcal{S}_2$ have the same number of left and right parentheses
- (Note that we can't assign a specific length to $\mathcal{S}_1$ or $\mathcal{S}_2$. We just know that the length of each is less than k. That's why we use complete induction rather than ordinary induction in this case)

Finishing the Induction Step

- So the total number of right parentheses in $\mathcal{S}$ is:

  1 + (number of right parentheses in $\mathcal{S}_1$) + (number of right parentheses in $\mathcal{S}_2$)
- And the total number of left parentheses in $\mathcal{S}$ is:

  1 + (number of left parentheses in $\mathcal{S}_1$) + (number of left parentheses in $\mathcal{S}_2$).
- These two numbers must be equal, since they are sums of three numbers, each number equal to its counterpart in the other sum
- This completes the induction step, and so the proof.

Remarks of a common-sensical nature

- Of course, what we just proved by induction is a piece of common sense:
- *Of course* there will be the same number of left as right parentheses, since the only way for a parenthesis to get into a formula is to get there by an application of the recursion clause, and the recursion clause just adds exactly 1 of each type of parenthesis, every time.
- The point is not that this is a hard problem but rather that it is so simple that it helps make clear how common-sensical the inductive pattern of reasoning is, when it isn't bound up with other complications

Wise Words, from our hero Frege

- Gottlob Frege—who worked out most of the core ideas in the modern approach to logic—wrote in 1884:
  If, by examining the simplest cases, we can bring to light what humankind has there done by instinct, and extract from such procedures what is universally valid in them, may we not thus arrive at general methods for forming concepts and establishing principles that will be applicable in more com- plicated cases? (Foundations of Arithmetic §2)
- And that's what we've often done above: take common sense cases and extract a general pattern we can use in more complicated cases, where the answers are not so obvious.
- Sensing strong Descartes-energy here

3. Mathematical Induction & Recursive Definitions

g. More induction and recursion on strings

aab strings: a common sense version

- Another example of a proof by induction so simple that you might think it silly to spell it out
- We deal again with a “language” we know and love: all strings of letters from the alphabet $\{a, b\}$.
- Call a string in this language an ‘aab−string’ if it consists of nothing but a series of the letters aab repeated some number of times.
- So aab, $aabaab$, $aabaabaab$, $aabaabaabaab$, . . . are aab−strings.

A Radical Claim

- I now claim that any aab–string has more a's in it than b's
- Your reaction might be "Can you believe this guy? Back at it again with the trivial examples. *Of course* there are more a's than b's in an aab–string."
- "In fact, there will always be exactly twice as many a's as b's. That's *obvious* from the definition!"
- When you say this, you are implicitly using recursive/inductive reasoning

Heuristic Recursive Reasoning (not a proof!)

- We spell out what is implicit in the “it’s obvious” reaction. Sometimes our common-sense reactions can have more hidden logical intricacy than we realize
- (often, things are not what they seem. Consider the dark underbelly of suburbia)
- One way to look at it is this: You get *aab* strings by repeatedly sticking *aab*’s together
- Each time you stick on a new *aab* you add one *b* and two *a*’s. And you start out with more *a*’s than *b*’s in *aab*, the shortest *aab*-string.
- There’s no way to get more *b*’s than *a*’s with that process, no matter how many times you repeat it.

Translating common sense into an inductive argument

- To translate the common sense reaction, we first make explicit—using recursion!—the idea that aab–strings are built up by successively adding aab:
  1. **Base clause**: aab is an aab–string
  2. **Recursion clause**: If s is an aab–string, then $s * aab$ is an aab–string
  3. **Closure clause**: Nothing else is an aab–string in $\{a, b\}$
- Next, use the fact that the aab–strings are built up this way to argue **by complete induction on string length** that they must have more a's than b's

Starting the Inductive Proof on *aab*−strings

- **Base Case**: *aab* has more *a*'s than *b*'s
- **Induction Step**: *Assume* that for any arbitrary *aab*−string *s* with fewer than *k* letters (where $(3 < k)$), the property holds, i.e. that this string *s* has more *a*'s than *b*'s.
  - this is our 'Induction Hypothesis' for Complete Induction
  - $**$Note – we need to make *k* greater than 3, because the string in the base case has 3 letters

- Try to finish the proof yourself before flipping to next slide!
  Hint: consider an arbitrary *aab*−string *t* with *k* letters.

Finishing the Induction Step for aab–strings

- Let t be an arbitrary aab–string with exactly k letters
  - Since $3 < k$, that means that t cannot be aab, so it must have been produced by some applications of the recursion clause
  - That means that there is some other aab–string t', such that $t = t' * aab$.
- t' is three letters shorter than t so it has fewer than k letters
  - Since it has fewer than k letters it falls within the scope of the induction hypothesis, and so it has more a's than b's.
  - Since t has two more a's and only one more b than t', that means t has more a's than b's as well.
- This completes the induction step and hence the proof

3. Mathematical Induction & Recursive Definitions

h. Extra ‘Big Picture’ stuff for Recursion

Defining ‘Definition’

- What do we want a definition to do?
- We want to give meaning to a word that previously had no meaning, and we want it to do it in a way that allows a person who didn’t previously know what the word meant to come to know what it means.
- Simple example: I introduce a new word into our vocabulary: grunsk. What does it mean? Well, something is grunsk if and only if it is a blue volleyball.
- OK, you might not see much point to having a word like this, but the definition does allow you to use it. All you need to now is what “ is a blue volleyball” means, and you can confidently divide the world into the grunsks and not-grunsks.

Ways to #fail: e.g. Circular Definitions

- There are ways that this pattern can screw up: it might be that the words are too vague, or perhaps they don’t pick out a unique thing or class.
- Another way a definition can fail is if it is circular, such as: “Something is grunsk if and only if it is a blue grunsk.”
- That definition doesn’t do what it’s supposed to do.
- Say I hand you a blue thing and ask you if it is a grunsk. You can’t use the definition to decide unless you already know what a grunsk is.
- Let’s say you hear this and you say to yourself: OK I’ve been warned. No circular definitions for me, thank you very much.
- How to spot them so you can avoid them?

A Guideline to Avoid Circularity

- This looks like a good guideline: set the word you want to define on a specific side of the definition (usually it’s the left hand side, but our recursive definition has the word to be defined on the right). Then if the word also occurs on the *other* side, the definition is circular.
- And usually, that’s a trustworthy guideline. But not in the case of recursive definitions. In this case things are more subtle.
- I’ll continue with the definition of our SL wffs and then revisit the point.

The Base Clause

- First the Base clause: Every atomic sentence is a wff/sentence.
- You might think there is some circularity here: Isn’t the word “sentence” on both the right and the left side, at least when we call ‘wffs’ ‘sentences’?
- But “atomic sentence ” is a defined expression: we defined it by listing the capital letters of the English alphabet, allowing subscripts from the natural numbers
- You don’t need to know what the word “sentence” means to understand what an atomic sentence is from this definition.

The Whole as Distinct from its Parts

- It can happen that we understand what an expression as a whole means, even if we don't understand the parts
- (Most of us understand what "hoist with his own petard" means, (like many clichés, this is a turn of phrase originally introduced by Shakespeare) even though we misunderstand the relevant meaning of "hoist", and we don't know what a "petard" is.)
- In the words of the philosopher Quine: such occurrences of words are like "cat" in "cattle" – you don't have to understand "cat" to know what cattle are.
- You don't have to know what "sentence" means to understand the stipulation: The *atomic sentence letters* are the capital Roman letters in the English alphabet, with or without number subscripts.

3. Mathematical Induction & Recursive Definitions

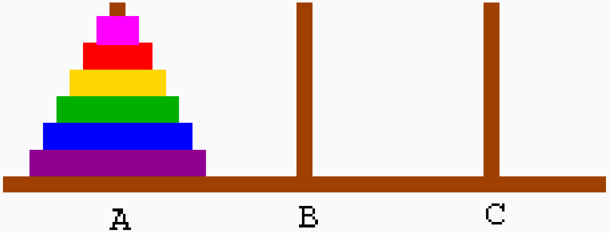
i. Towers of Hanoi Example

Towers of Hanoi

- As a concrete, non-mathematical example, we'll consider a puzzle that looks at first as if it might be very hard, but it becomes simple and straightforward if we think about it in terms of recursion.
- This is a tabletop puzzle called the **Towers of Hanoi**.
- You are given three pegs and n discs. No two discs are the same size.
- At the beginning of the game, all n discs are all on the leftmost peg, with the largest disc on the bottom, and then the remaining discs in decreasing order of size piled on top of the first, with the smallest on top.

Towers of Hanoi: Initial State

- Given three pegs and *n* discs. No two discs are the same size.
- Initial state: all *n* discs are all on the leftmost peg, with the largest disc on the bottom, and then the remaining discs in decreasing order of size piled on top of the first, with the smallest on top.
- Here is a drawing of the opening setup in the case of 6 discs



Towers of Hanoi: Goals and Rules

- Goal: end up with all discs on the right peg, in the same order, that is, smallest on top, and increasing in size towards the bottom.
- There are restrictions on how the discs are moved:
  1. The only allowed type of move is to take **one** disc from the top of a pile and drop it on another peg.
  2. Moving several discs at once is not allowed.
  3. A larger disc can never lie above a smaller disc, on any peg.

Towers of Hanoi

- Question: Is it possible to solve the puzzle for any number of discs?
- It turns out that by thinking recursively, we can find a very simple answer.
- If you haven't already read the notes, try experimenting with the puzzle to see what you think.
- There is an online version at Math is Fun!

4. Truth Trees

4. Truth Trees

a. Why Trees?

Why Trees?

- Truth tables become a bit tedious to work with
- We would like a more streamlined method for checking INconsistency (i.e. UNsatisfiability) and validity
- Trees are often faster and have less ‘irrelevant’ information

Why not ‘Partial Truth Tables’ (PTTs)?

- Couldn’t we just look at the ‘relevant’ lines in a truth table?
- e.g. are the following sentences consistent?
- $P \mathbin{\&} Q$; $\sim(P \supset \sim Q)$; $(S \mathbin{\&} R) \vee \sim J$
- What about $P \mathbin{\&} Q$; $\sim(P \supset \sim Q)$; $(S \mathbin{\&} R) \vee \sim J$; $\sim(S \vee R) \mathbin{\&} J$
  - Showing this formally would require a 32 row truth table...
  - But the answer is clear from reasoning about truth conditions

Where Partial Truth Tables shine

The method of partial truth tables is already rigorous in cases where a single truth value assignment (TVA) suffices for an answer:

1. Consistent set of sentences:
    - Sufficient to find one TVA on which each sentence is true
2. Invalid argument:
    - Find one TVA where premises are true but conclusion is false
3. Logically inequivalent sentences:
    - Suffices to find one TVA where they differ in truth value

Where Partial Truth Tables Falter

So far, our only rigorous method for answering some questions *requires* a complete truth table, even when the answer is ‘obvious’

1. INconsistent set of sentences:
    - Need to show that NO TVA makes every sentence true
2. Valid argument:
    - Need to show there is NO TVA where the premises are true but the conclusion is false
3. Logically equivalent sentences:
    - Need to show that the sentences agree on EVERY TVA

Motivation for Trees (System STD)

- Trees formalize our pattern of thinking when making PTTs
  - For proofs of consistency, invalidity, or inequivalence, there is really little difference
  - Trees basically ‘keep track of our mental work’

- But for validity, inconsistency, and logical equivalence, we really get something new:
- Trees formalize our ‘shortcut’ arguments without needing to consider every TVA to the relevant atomic sentences
- (Although to be fair, the rigor of this ‘shortcut’ is beholden to our soundness result. But that’s work you do once and then have FOREVER—much like a diploma!)

Similarities with Truth Tables

- Like truth tables, trees are *mechanical*:
  - No insight or creativity required (woooooooo!)
  - Just execute the resolution ‘algorithm’
- Trees give equivalent answers to truth tables
  - Will prove this soon (soundness and completeness of STD)

What we will do (soon!) to Demonstrate 'Rigor'

- By proving that our tree system is *sound*, we show that these shortcut arguments are rigorous (they never lead us astray)
  - **Sound**: Single turnstile entails Double Turnstile
  - (syntactic to semantic: i.e. we chose 'good' rules!)
- By proving that our tree system is *complete*, we will show that we never need truth tables: trees suffice
  - **Complete**: Double Turnstile entails Single turnstile
  - (semantic notions are fully covered by our syntactic rules)
  - (Means: we wrote down *enough* rules!)

4. Truth Trees

b. Tree Rules (trees rule!)

Sentential Tree Derivations (STD)

- The method of trees is our first derivation system
- To distinguish it from our later natural deduction (ND) systems, we will call this system ‘Sentential Tree Derivations’ (STD)!
- As a proof system, it comes with its very own single turnstile:

$$\vdash_{STD}$$

- As a proof system, it is *purely syntactic*: it is defined entirely in terms of legal rules, with no explicit mention of truth or falsity

Whence these rules?

- For negation, we have a single rule: double negation elimination
- For each binary connective, we have two rules:
  one for an unnegated sentence; one for a negated sentence
- Where do these rules come from?
- They come from the truth conditions for the connectives
- If you think about what it means for a formula to be true, you can always derive the rules

Stacking vs. Splitting (aka ‘branching’)

- There are two basic kinds of rules,
  coming from conjunction and disjunction:
- Stacking: resolve $A \,\&\, B$ into ‘A stacked on B’
  - Idea: for a conjunction to be true, both conjuncts must be true
- Splitting: resolve $A \vee B$ into an A–branch and a B–branch
  - Idea: for a disjunction to be true, either disjunct must be true
- Atomic formulae and their negations can’t be further resolved

Your Very First Tree

- Use a tree to determine whether the following sentences are consistent (i.e. jointly satisfiable):

$$\sim A \,\&\, \sim D;\ C \,\&\, (B \vee A);\ \sim B \vee C;\ \sim D \,\&\, \sim F$$

- Some things to NEVER forget!
  - Each formula gets its own line, replete with line NUMBER
  - Justify new ‘nodes’ by citing the line number of a formula you are ‘resolving’; note the rule you applied in the far right ‘column’
  - Put a check ‘✓’ next to a formula once you resolve it
  - (Perhaps put a colon ‘:’ before each justification, in order to get used to what *Carnap* requires for natural deduction, e.g. :3 &)

The Rules in Themselves

Let’s go through the rules!

1. Double negation (elimination)
2. Conjunction and Negated conjunction
3. Disjunction and Negated disjunction
4. Conditional and Negated conditional
5. Biconditional and Negated biconditional
    - Motivation: $A \& B$ branch or $\sim A \& \sim B$ branch

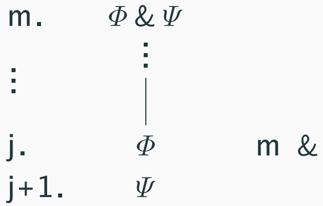
Double Negation

Double Negation ($\sim$)

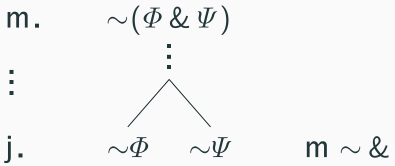
$$\begin{array}{lll} \text{m.} & \sim\sim\Phi & \\ \vdots & \vdots & \\ & | & \\ \text{j.} & \Phi & \text{m} \sim \end{array}$$

Conjunction and Negated Conjunction

Conjunction (&)

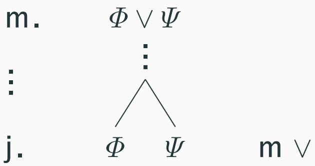


Negated Conjunction ($\sim$ &)

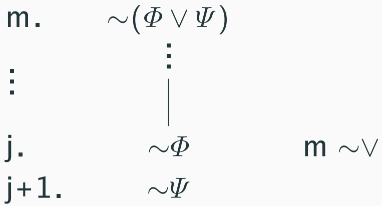

m. $\sim(\Phi \,\&\, \Psi)$
⋮
j. $\sim\Phi$ $\sim\Psi$ m $\sim$ &

Disjunction and Negated Disjunction

Disjunction (∨)



Negated Disjunction (∼∨)



Conditional and Negated Conditional

Conditional (⊃)

$$\begin{tikzcd}[row sep=small]
\text{m.} & & \Phi \supset \Psi \arrow[d, phantom, "\vdots"] & & \\
\vdots & & {} \arrow[dl, no head] \arrow[dr, no head] & & \\
\text{j.} & \sim\Phi & & \Psi & \text{m} \supset
\end{tikzcd}$$

Negated Conditional (∼⊃)

$$\begin{tikzcd}[row sep=small]
\text{m.} & \sim(\Phi \supset \Psi) \arrow[d, phantom, "\vdots"] & \\
\vdots & {} \arrow[d, no head] & \\
\text{j.} & \Phi & \text{m} \sim\supset \\
\text{j+1.} & \sim\Psi &
\end{tikzcd}$$

Biconditional and Negated Biconditional

Biconditional ($\equiv$)

$$\begin{array}{llll}
\text{m.} & \Phi \equiv \Psi & & \\
\vdots & \vdots & & \\
 & \swarrow \quad \searrow & & \\
\text{j.} & \Phi \qquad \sim\Phi & & \text{m} \equiv \\
\text{j+1.} & \Psi \qquad \sim\Psi & &
\end{array}$$

Negated Biconditional ($\sim\equiv$)

$$\begin{array}{llll}
\text{m.} & \sim(\Phi \equiv \Psi) & & \\
\vdots & \vdots & & \\
 & \swarrow \quad \searrow & & \\
\text{j.} & \Phi \qquad \sim\Phi & & \text{m} \sim\equiv \\
\text{j+1.} & \sim\Psi \qquad \Psi & &
\end{array}$$

4. Truth Trees

c. Grow your own Trees!

Planting Trees

The Root of the Tree

You begin a tree by writing a list of formula(s)

- Every formula sits on its own line number
- This list is called the **root** of the tree

(The list is finite and non–empty; contains at least one formula)

- Grow your tree by adding **nodes**
- We count the root as a 'node' as well
- Each non–root node contains *one or two* wffs (per the rules)
  each new node is connected to the node above it
- Introduce a new line number for every new (row of) sentence(s)
  (line numbers correspond to (rows of) sentence(s), NOT nodes)

Where do the non-root nodes come from?

- We **resolve** sentences!
  (i.e. break them down, one main connective or negated formula at a time—much like our New Year’s Resolutions)
- We start by resolving the sentences in the root
- We then resolve any new nodes created, and so on
- Until we hit rock bottom! i.e. atomic formulae or their negations
- (Sometimes you can stop before resolving all sentences)

Closed vs. Open Branches

- A **branch** is a path taking you from the root through a series of nodes, each connected to the one above it
- A branch is **closed** if a wff and its negation appear in its nodes
    - **We write a** ‘$\times$’ beneath each closed branch
    - Closure results from ANY wff and its negation (not just atomic)
- Otherwise, a branch is **open**
    - As you grow your tree, any branch that is not closed is open
    - We are particularly interested in branches that *remain open* even after every formula has been resolved
    - These branches are **complete** and **open**

Completing a Branch

- A branch is **complete** when every resolvable formula appearing on it has been resolved (you've then 'completed' the branch)
- i.e., we have applied the tree rules until nothing but atomic formulae or their negations appear (remember: inclusive 'or'!)

- In a **complete open branch**, it is
  - (i) not the case that both a sentence and its negation appear in its nodes (i.e. no contradictions along the branch)
  - and (ii) all formulae on the nodes have been resolved (so there's no possibility of a contradiction appearing later)
  - **We write an** '$\uparrow$' beneath each complete open branch

- *Psssst, semantic point!*: a complete open branch indicates a TVA that makes each of the sentences in the root true

Returning to our Earlier Examples

Let's use trees!

- e.g. are the following sentences consistent?
- $P \,\&\, Q$; $\sim(P \supset \sim Q)$; $(S \,\&\, R) \vee \sim J$
- What about $P \,\&\, Q$; $\sim(P \supset \sim Q)$; $(S \,\&\, R) \vee \sim J$; $\sim(S \vee R) \,\&\, J$
- Question (for later): can we reinterpret this second result as a valid argument? What are the premises? What is the conclusion?

Warning: No Logical Equivalencies!

- Keep in mind that the rules of STD are entirely syntactic
- You can use ONLY these nine rules and no others
- In particular, STD itself knows nothing about logical equivalence
- So you CANNOT replace a sentence willy-nilly with a logically equivalent one, unless this is sanctioned by one of our rules
- Likewise, you can close a branch only if some wff Φ and its negation $\sim\Phi$ appear in the branch

4. Truth Trees

d. Using Trees

Syntactic equivalents of our Semantic Notions

- Recall that we are interested in assessing various semantic properties of sentences of SL:

  1.) Contradiction? Tautology? Logically Contingent?
  2.) Equivalent? Inequivalent?
  3.) Consistent? Inconsistent?
  4.) Valid argument/entailment? Invalid argument/not entailed?

- For each semantic notion, there is a corresponding syntactic property of a tree
  - (Although one has to prove this correspondence exists)

Tree-contradictions and Tree-tautologies

Tree-contradiction

- A sentence Φ is a tree-contradiction if there is a tree that starts with Φ that has only closed branches
- (definition only requires the existence of a single such tree)
- (it says nothing about *all* trees starting with Φ)

Tree-tautology

- A sentence Ψ is a tree-tautology if there is a tree that starts with $\sim\Psi$ that has only closed branches
- i.e., provided that $\sim\Psi$ is a tree-contradiction
- In this case, we write '$\vdash_{STD} \Psi$'
- (again: this definition requires the existence of single such tree)

A limitation of these syntactic definitions

- Notice that the definitions of ‘tree-contradiction’ and ‘tree-tautology’ leave open the possibility that a sentence could be both a tree-contradiction and a tree-tautology
- This is because each definition requires the existence of only a single tree with a given syntactic property
- Obviously, if system STD is any good, this won’t be possible! But we’ll need to prove this (perhaps on PS 5)!
- Taking for granted that system STD is ‘good’, any tree-contradiction is a contradiction, and any tree-tautology is a tautology

Tree-consistency

Question: when is a set Γ of wffs consistent (i.e. jointly satisfiable)?

- Construct a tree whose root is all sentences in Γ
- Next, apply the tree-rules until either

  1.) **Each branch closes**, in which case the argument is **tree-inconsistent**

  2.) You have a complete open branch, in which case the argument is **tree-consistent**

- (Pssst semantic aside: the complete open branch indicates a truth value assignment that makes each sentence in Γ true)

Connections between consistency and validity

- Recall from long ago: an argument is **valid** if and only if the premise set and the negation of the conclusion is **inconsistent** (i.e. if the premises are true, the conclusion is not false)

- An argument is invalid if and only if the premise set and the negation of the conclusion is consistent (i.e. the premises and the negated-conclusion are satisfiable)

- These connections motivate our definitions of tree-validity and tree-invalidity

Tree-valid vs. Tree-invalid

- Consider an argument with premises given by a set Γ of wffs and conclusion Φ (i.e. Γ could be multiple sentences)
- Construct a tree with the following root: all sentences in Γ along with $\sim\Phi$ (i.e. the NEGATION of the conclusion)
- Next, apply the tree-rules until either

1.) **Each branch closes**, in which case the argument is **tree-valid**

- In this case, we write $\Gamma \vdash_{STD} \Phi$

2.) You have a complete open branch, in which case the argument is **tree-invalid**

Using trees to check for Validity

Since most homework problems follow this pattern, let's make it really explicit!

1. Add each premise to the root (number each line)
2. Add the **NEGATION** of the conclusion to the root
3. Resolve sentences until either:
    - **Each branch closes**, in which case the argument is **valid**
    - You have a complete open branch $\Rightarrow$ the argument is **invalid**

Don't forget to **justify each new node** by citing the line you are resolving and the rule you are applying

Remember that a branch closes whenever a sentence and its negation appear in its nodes (these need not be atomic sentences)

Using trees to check Tautologies

Likewise for whether a sentence is a tautology:

1. Add the **NEGATION** of the sentence to the root
2. Resolve sentences until either:
   - **Each branch closes**, in which case the sentence is **tautologous** (semantic aside: it's impossible to make the sentence false)
   - You have a complete open branch, in which case the sentence is NOT a tautology
     (semantic aside: it is possible to satisfy the sentence's negation, so it's possible to make the sentence in question false)

Never forget to justify! Always remember!

- Just to repeat something you are liable to forget to do:

- Never forget to **justify each new node** by

  1.) citing the line you are resolving (e.g. ‘3’) and

  2.) citing the rule you are applying (e.g. ‘$\sim\vee$’)

  These justifications go in the ‘rightmost column’

Another Eternal Memory!

- If your tree has branched and you are resolving a wff, you have to put the result under EVERY branch connected to the wff
- More precisely: To resolve a sentence, you have to extend **ALL** of the open branches *running through the node* of the given wff
- Example: consider a root with $A \vee B$ and $P \& Q$. Check what happens when you resolve '$A \vee B$' first, followed by '$P \& Q$'
- If a branch is already closed, you don't have to worry about it

4. Truth Trees

e. Topical Topiary Tips

How to Sculpt a Tree

- With trees, as with life, you’ve got options!
- You can resolve sentences in any order you please
- But some resolution orders will be faster/easier/more convenient than others (they’ll at least involve ‘less ink’, and someone is paying for that ink!)
- Corporate America and BigPharma want you to SAVE INK!

Rules of thumb for Green thumbs

1. Temporally favor ‘stacking’ rules over ‘splitting’ rules

2. Given a choice, resolve sentences that do not lead to new open branches:

    - Save splitting until you’ve closed as many branches as possible

    - Otherwise, you can end up with a lot of branches!

      $\Rightarrow$ and that’s bad topiary!

An Example to Work Through

Let us illustrate these morals, since one burnt by the flame fears fire for life:

- Is the argument from $C \supset P$, $P \vee D$, $\sim(Q \equiv C)$ to D valid?
- In the worst tree, resolve $C \supset P$, then $\sim(Q \equiv C)$, and then $P \vee D$
- In the best tree, resolve $P \vee D$, then $C \supset P$, and then $\sim(Q \equiv C)$
- Often we should take the road most traveled, and that will make all the difference

Our Running Example

Sarah lives in Chicago or Erie.
Amir lives in Chicago unless he enjoys hiking.
If Amir lives in Chicago, Sarah doesn't.
Neither Sarah nor Amir enjoy hiking.
∴ Sarah lives in Erie.

$$C \vee E$$

$$A \vee M$$

$$A \supset {\sim}C$$

$${\sim}S \,\&\, {\sim}M$$

$$\therefore E$$

Let us tree!

Our Running Example (simplified)

Recall that to handle this with a truth-table, we simplified the last premise (to eliminate 'S' and avoid a 32 row truth table):

Sarah lives in Chicago or Erie.
Amir lives in Chicago unless he enjoys hiking.
If Amir lives in Chicago, Sarah doesn't.
Amir doesn't enjoy hiking.
$\therefore$ *Sarah lives in Erie.*

$$C \vee E$$
$$A \vee M$$
$$A \supset {\sim}C$$
$${\sim}M$$
$$\therefore E$$

A	C	E	M	$C \vee E$	$A \vee M$	$A \supset \sim C$	$\sim M$	E
T	T	T	T	T T T	T T T	T F F T	F T	T
T	T	T	F	T T T	T T F	T F F T	T F	T
T	T	F	T	T T F	T T T	T F F T	F T	F
T	T	F	F	T T F	T T F	T F F T	T F	F
T	F	T	T	F T T	T T T	T T T F	F T	T
T	F	T	F	F T T	T T F	T T T F	T F	T
T	F	F	T	F F F	T T T	T T T F	F T	F
T	F	F	F	F F F	T T F	T T T F	T F	F
F	T	T	T	T T T	F T T	F T F T	F T	T
F	T	T	F	T T T	F F F	F T F T	T F	T
F	T	F	T	T T F	F T T	F T F T	F T	F
F	T	F	F	T T F	F F F	F T F T	T F	F
F	F	T	T	F T T	F T T	F T T F	F T	T
F	F	T	F	F T T	F F F	F T T F	T F	T
F	F	F	T	F F F	F T T	F T T F	F T	F
F	F	F	F	F F F	F F F	F T T F	T F	F

Every valuation makes at least one premise false, or makes the conclusion true: the argument is valid.

A fun Question

- Question: how can you use a single tree to show that two sentences are logically equivalent?
- Answer is on next slide! No spoilers!
- Example: show that $P \vee Q$ is logically equivalent to $\sim\Big((\sim(P \equiv Q)) \equiv (P \mathbin{\&} Q)\Big)$
- (where this represents $(x + y) + (x \cdot y)$ in our Boolean algebra)

Answer to our ‘fun Question’

- Question: how can you use a single tree to show that two sentences are logically equivalent?
- Answer:
- Consider whether the biconditional of the two formulae is a tautology
- So make a tree whose root is the negation of this biconditional
- If all branches close, then this negation is unsatisfiable, i.e. is a contradiction. In which case the biconditional is a tautology. In which case the two wffs are logically equivalent.

4. Truth Trees

f. Practice with Proofs

Two Questions about (semantic) Entailment

Two questions we never got around to answering:

Recall: '$\Gamma \nvDash \Psi$' means that the (set of) sentence(s) Γ does not semantically entail Ψ, i.e. an argument from Γ to Ψ is invalid.

1. True or False? If $\Phi \vDash \Psi$, then $\sim\Phi \nvDash \Psi$

2. True or False? If $\Gamma \vDash \Phi$ and $\Delta, \Phi \vDash \Psi$, then $\Gamma, \Delta \vDash \Psi$?

And now with trees!

Let’s answer syntactic analogs of these questions in system STD:

Recall: ‘$\Gamma \nvdash_{STD} \Psi$’ means that arguing from Γ to Ψ is NOT tree-valid (and with soundness, this means it is tree-invalid)

1. True or False? If $\Phi \vdash_{STD} \Psi$, then $\sim\Phi \nvdash_{STD} \Psi$

2. True or False? If $\Gamma \vdash_{STD} \Phi$, then $\Gamma, \Delta \vdash_{STD} \Phi$

3. True or False? If $\Gamma \vdash_{STD} \Phi$ and $\Delta, \Phi \vdash_{STD} \Psi$, then $\Gamma, \Delta \vdash_{STD} \Psi$?

An Induction example because...why not?

Gotta stay sharp!

Prove the following by induction. Don't forget to explicitly state the base case and the induction step!

3. If a wff doesn't contain any binary connectives, then it is contingent.
   (hint: say that a wff is *baller* if it either contains a binary connective or is contingent. Use induction to show that every wff is baller.)

5. Metalogic for STD

5. Metalogic for STD

a. Big Picture Stuff

Trees as a Shortcut, provided that...

- As we have seen, trees provide a shortcut for demonstrating that a set of sentences is inconsistent (i.e. unsatisfiable): construct a tree whose root is these sentences s.t. all branches close
- Underwrites further shortcuts for demonstrating:
  1.) that an argument is valid
  (its premises and negated conclusion are unsatisfiable)
  2.) that a sentence is a tautology (its negation is unsatisfiable)
  3.) that two sentences are logically equivalent
- But our shortcuts are justified only if system STD is *sound*
- And guaranteed to have a shortcut only if system STD is *complete*

A Tale of Two Turnstiles: the syntactic one

- Recall that the single turnstile ‘$\vdash_{STD}$’ stands for provability (aka derivability) within our proof system STD
- “$\Gamma \vdash_{STD} \Theta$” means that we can derive Θ from Γ, within STD. The argument with premises Γ and conclusion Θ is ‘**tree-valid**’
- Equivalently, this means that $\Gamma \cup \{\sim\Theta\}$ is **tree-inconsistent**: There is a tree with this set as the root s.t. **all branches close**
- Throughout, ‘Γ’ is a *finite* set of SL sentences

A Tale of Two Turnstiles: the semantic one

- Recall that the double turnstile '$\vDash$' stands for semantic entailment (aka logical consequence) within (classical) sentential logic SL.
- "$\Gamma \vDash \Theta$" means that Γ logically entails Θ
  Whenever the premises in Γ are true, the conclusion Θ is true
- Equivalently: there is no truth–value assignment (TVA) s.t. Γ is satisfied while Θ is false
- Equivalently, this means that $\Gamma \cup \{\sim\Theta\}$ is logically inconsistent: no TVA satisfies the premises and negated conclusion

Promises Made, Promises Kept

- By proving that our tree system is *sound*, we show that these shortcut arguments are rigorous (they never lead us astray)
    - **Sound**: If $\Gamma \vdash_{STD} \Theta$, then $\Gamma \vDash \Theta$
    - (syntactic to semantic: i.e. we chose 'good' rules!)

- By proving that our tree system is *complete*, we will show that we never need truth tables: trees suffice
    - **Complete**: If $\Gamma \vDash \Theta$, then $\Gamma \vdash_{STD} \Theta$
    - (semantic notions are fully covered by our syntactic rules)
    - (Means: we wrote down *enough* rules!)

Basic Proof Strategy

- Notice that both soundness and completeness are if–then statements (i.e. of the form $P \supset Q$):
  - **Sound**: If $\Gamma \vdash_{STD} \Theta$, then $\Gamma \vDash \Theta$
  - **Complete**: If $\Gamma \vDash \Theta$, then $\Gamma \vdash_{STD} \Theta$
- Hence, they are logically equivalent to their contrapositives:
  - Contrapositive of $(P \supset Q)$ is $({\sim}Q \supset {\sim}P)$
  - **Soundness**: If $\Gamma \nvDash \Theta$, then $\Gamma \nvdash_{STD} \Theta$
  - **Completeness**: If $\Gamma \nvdash_{STD} \Theta$, then $\Gamma \nvDash \Theta$
- We will prove the contrapositives, using induction! Wooooo!

5. Metalogic for STD

b. Soundness of System STD

Sounding out Soundness

- **Sound**: If $\Gamma \vdash_{STD} \Theta$, then $\Gamma \vDash \Theta$
- *Sound$_{contra}$*: If $\Gamma \nvDash \Theta$, then $\Gamma \nvdash_{STD} \Theta$
- As always: ask what this means, based on the definitions:
- "$\Gamma \nvDash \Theta$" means that *it is not the case that* the set Γ entails Θ, i.e. there is a TVA where Γ is true but Θ is false.
  - So on this TVA, $\sim\Theta$ is true $\Rightarrow \Gamma \cup \{\sim\Theta\}$ is CONSISTENT

Analyzing Key Definitions Continued

- *Sound*$_{contra}$: If $\Gamma \not\models \Theta$, then $\Gamma \nvdash_{STD} \Theta$
- "$\Gamma \nvdash_{STD} \Theta$" means that *it is not the case that* the argument from Γ to Θ is **tree-valid**
  - i.e., *it is not the case that* there exists a tree with root $\Gamma \cup \{\sim\Theta\}$ that possesses **all closed branches**
  - Equivalently: ANY tree with root $\Gamma \cup \{\sim\Theta\}$ possesses **at least one complete open branch**
  - (Aside: this is NOT the same as saying that the argument is **tree-invalid**, since that only requires the existence of a single tree with a complete open branch)

Putting these Two Pieces Together

- Assume $\Gamma \nvDash \Theta$, i.e. assume that $\Gamma \cup \{\sim\Theta\}$ is CONSISTENT
    - Then, there is a TVA that makes the premises true and the conclusion false. Call this TVA '$\mathcal{I}$', which we'll use throughout
- Need to Show: ANY tree with root $\Gamma \cup \{\sim\Theta\}$ possesses a complete open branch (i.e. the argument is NOT tree-valid)
- Equivalently, we are showing that if the root is consistent, then any tree with that root possesses a complete open branch
- (i.e. whenever the root is satisfiable, the tree will not close)

Key Facts about our TVA $\mathcal{I}$

- By definition, $\mathcal{I}$ makes Θ false but everything in Γ true
- As a TVA, $\mathcal{I}$ assigns either ‘true’ or ‘false’ (exclusive–or) to every atomic sentence letter appearing in the sentences of Γ and Θ
- Hence, $\mathcal{I}$ determines a truth value for any wff built from these atomic sentences
- for any such wff Ψ, $\mathcal{I}$ makes Ψ true if and only if it makes $\sim\Psi$ false

Some Convenient Definitions to Streamline our Inductive Proof

- Call a tree **good** if its root contains only $\sim\Theta$ and the wff in Γ
- Say that a branch in a good tree is an **$\mathcal{I}$–branch** if every wff on it is $\mathcal{I}$–true, i.e. true according to $\mathcal{I}$
- Note that each $\mathcal{I}$–branch is open, since $\mathcal{I}$ cannot make true both a wff and its negation (so an $\mathcal{I}$–branch can't be closed!)
- **Proof plan**: show that every **good** tree contains an **$\mathcal{I}$–branch**
- This will include all the completed good trees, which will have a *complete* open branch, which is what we want to show

Set-Up for Induction

- Trees have a recursive structure (they 'grow' by nine rules), $\Rightarrow$ we can perform proof by induction (woooooooooo)!
- We'll do complete induction over the *number of nodes*! (could also do ordinary induction on # of times we apply a rule)
- **Base case**: the smallest good tree (1 node)
- **Induction hypothesis**: Assume that every good tree with n-many nodes, where $1 \leq n < k$, contains an $\mathcal{I}$-branch
- Induction Step: show that an arbitrary good tree with k nodes also contains an $\mathcal{I}$-branch (where $k > 1$)

Getting this Party Started: the Base Case

- **Base case**: consider a good tree with one node
- This is just the root $\Gamma \cup \{\sim\Theta\}$
- By definition, the TVA $\mathcal{I}$ satisfies the root
- $\Rightarrow$ root is an $\mathcal{I}$–branch, i.e. every wff on it is true according to $\mathcal{I}$

Keeping the Party Going

- Consider an arbitrary good tree 'Theodore' with k nodes ($k > 1$)
- Then there must be a good tree 'Theo' with $1 \leq n < k$ nodes such that 'Theodore' results from applying one of our nine STD rules to 'Theo'. Call this rule 'Ruby' $\in \{\sim, \&, \vee, \supset, \equiv, \sim\&, \sim\vee, \sim\supset, \sim\equiv\}$
- Since 'Theo' falls within the scope of our induction hypothesis, 'Theo' has at least one $\mathcal{I}$–branch, called 'Ida'
- Case a) Ruby extends a different open branch than Ida. Then Theodore still has $\mathcal{I}$–branch Ida, and so has an $\mathcal{I}$–branch
- Case b) Ruby extends Ida into branchlet(s). We need to show that for any rule, at least one of the branchlets is an $\mathcal{I}$–branch

The Simple way of Putting Case b)

- We have to show the following: if a good tree Theo contains an $\mathcal{I}$–branch, then so do all the trees that can be built from it by applying a single tree rule
  (all the possible Theodore's, from all the possible Ruby's)
- i.e., each of our nine rules preserves the property of having at least one $\mathcal{I}$–branch (i.e. a branch that satisfies the root)
- The book's induction step basically begins here at 'Case b'. We've just justified why it's okay to jump right to Case b.
- This mirrors our lazy induction schema for SL: where we start with two arbitrary wffs Φ and Ψ that have the property, and show that any application of the SL recursion clause preserves the property

Subcase i) Ruby is Double Negation ($\sim$)

- Suppose Theo's $\mathcal{I}$–branch Ida contains $\sim\sim\Psi$
- Ruby extends Ida to 'Idaho' by adding Ψ
- By assumption, $\sim\sim\Psi$ is true according to $\mathcal{I}$

  $\Rightarrow$ Ψ is also true according to $\mathcal{I}$

  - (Since otherwise, $\sim\Psi$ would be $\mathcal{I}$–true, but then $\sim\sim\Psi$ would be $\mathcal{I}$–false, which would contradict our assumption)
  - So Idaho is also an $\mathcal{I}$–branch, this time of Theodore

- Hence, our beloved Theodore contains an $\mathcal{I}$–branch

Subcase iii) Ruby is Negated Conjunction ($\sim\&$)

- Suppose Theo's $\mathcal{I}$–branch Ida contains $\sim(\Phi \,\&\, \Psi)$
- Ruby extends Ida by splitting her into a branchlet with $\sim\Phi$ and a branchlet with $\sim\Psi$
- By assumption, $\sim(\Phi \,\&\, \Psi)$ is true according to $\mathcal{I}$
  - So $\mathcal{I}$ cannot assign 0 to both $\sim\Phi$ and $\sim\Psi$, for if it did, then it would assign 1 to both Φ and Ψ, and hence 0 to $\sim(\Phi \,\&\, \Psi)$ (which would contradict our assumption)
  - So $\mathcal{I}$ must make at least one of $\sim\Phi$ or $\sim\Psi$ true
  - So **at least one** of our two branchlets must be an $\mathcal{I}$–branch
- Hence, Theodore contains at least one $\mathcal{I}$–branch

5. Metalogic for STD

c. Testing Alternative Rules: Soundness

Modifying system STD

- What if we had chosen an alternative rule(s)?
- Simplest case: we swap out one of our nine rules for a different rule, i.e. for a given connective or negated connective.
- Call our modified system ‘STD^*’
- Would our modified system STD^* remain Sound?
- Would our modified system STD^* remain Complete?

Checking Soundness: “top-down” reasoning

- **Heuristic for Soundness checking**: consider an **arbitrary** truth value assignment (TVA) that satisfies the sentence being resolved (i.e. *makes true* the sentence ‘at the top’ of the rule)
- Ask whether this TVA guarantees that **at least one** node below is satisfied, i.e. the sentences on that node are all made true
- If ‘**yes**’, then the rule preserves soundness: proceed to extend our soundness proof for this case (reasoning in terms of a TVA ‘$\mathcal{I}$’).
- If ‘**no**’, then the rule BREAKS soundness:
  proceed to **construct a counter-example** using the rule.

Counterexamples to Soundness

- To show soundness fails, it is NECESSARY to provide a CONCRETE counterexample, using SL sentences
  - Further heuristic reasoning about TVAs is not enough!

- What you need for a counterexample to soundness:
  1.) Choose a satisfiable root, i.e. a **consistent** set of sentences
  2.) Apply the modified rule in the tree; you may also use other rules that we've already shown preserve soundness
  3.) Show that the tree **closes** (i.e. NO complete open branches)

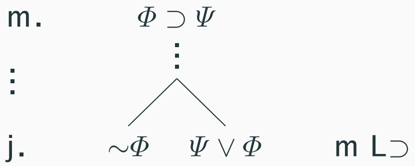
Why such Counterexamples work

- **Sound**: If $\Gamma \vdash_{STD^*} \Theta$, then $\Gamma \vDash \Theta$
- Counterexample: $\Gamma \vdash_{STD^*} \Theta$ but $\Gamma \nvDash \Theta$:
  - '$\Gamma \vdash_{STD^*} \Theta$' means tree with root $\Gamma \cup \{\sim\Theta\}$ **CLOSES**
    (i.e. is tree–valid in system STD^*)
  - '$\Gamma \nvDash \Theta$' means that $\Gamma \cup \{\sim\Theta\}$ is **CONSISTENT** (i.e. satisfiable)
    (i.e. there exists a TVA that makes the premises Γ true
    but the conclusion Θ false)
- If our system *were* sound, then whenever a tree closes, it *would* correspond to a semantically valid argument.

Liberal Conditional and Conservative Biconditional

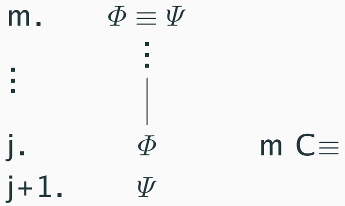
STD*: replace rule ($\supset$) with:

Liberal Conditional (L$\supset$)



STD†: replace rule ($\equiv$) with:

Conservative Biconditional (C$\equiv$)



Liberal Conditional: Heuristic Reasoning

- The following is ‘heuristic reasoning’ using our rule; it is NOT a formal answer to the question:
- Reason from the top–down: $\Phi \supset \Psi$ is logically equivalent to $\sim\Phi \vee \Psi$
- The sentences across the two nodes are logically equivalent to $(\sim\Phi) \vee (\Psi \vee \Phi)$
- So note that the top entails the (disjunction of the) bottom! So given an interpretation that satisfies $\Phi \supset \Psi$, it must satisfy the sentence in at least one branch below.
- Next, proceed to formally extend our soundness proof!

Liberal Conditional: Formal Answer

- **Formally**: assume $\Phi \supset \Psi$ is true according to $\mathcal{I}$.
- Then $\mathcal{I}$ assigns 0 to Φ or 1 to Ψ (or both).
- In the first case, $\mathcal{I}$ assigns 1 to $\sim\Phi$, satisfying the left branch.
- In the second case, $\mathcal{I}$ makes $(\Psi \vee \Phi)$ true, so it satisfies the right branch.
- Either way, $\mathcal{I}$ satisfies the new sentences on **at least one** branch.
- Hence, Liberal Conditional does not break soundness

Conservative Biconditional

- Reason from the top-down:
  $\Phi \equiv \Psi$ is logically equivalent to $(\Phi \,\&\, \Psi) \vee (\sim\Phi \,\&\, \sim\Psi)$
- The node below is logically equivalent to $(\Phi \,\&\, \Psi)$
- Note that the top does NOT entail the bottom! Given an interpretation that satisfies $\Phi \equiv \Psi$, it is NOT guaranteed to satisfy the sentence(s) in at least one branch below.
- Hence, **to the counterexample!**
  (note that the problem REQUIRES this! can't stop won't stop!)

Counterexample to Soundness of $STD^{\dagger}$

- For a counterexample, need: $\Gamma \vdash_{STD^{\dagger}} \Theta$ but $\Gamma \not\models \Theta$

  $STD^{\dagger}$: replace rule ($\equiv$) with *Conservative Biconditional* (C$\equiv$):

1.	$P \equiv P$	Premise (PR)
2.	$\sim P$	$\sim$Conclusion
	$\vert$	
3.	P	1 C$\equiv$
4.	P	
	$\times$	

  2, 3, so tree-valid in $STD^{\dagger}$

- But $(P \equiv P) \not\models P$ because $\{(P \equiv P), \sim P\}$ is consistent! Assign ‘P’ false! N.B.: your counterexample MUST use actual sentences of SL; not meta-variables!

5. Metalogic for STD

d. Completeness of System STD

Basic Proof Strategy for Completeness

- By proving that our tree system is *complete*, we will show that we never need truth tables: trees suffice
  - **Complete**: If $\Gamma \vDash \Theta$, then $\Gamma \vdash_{STD} \Theta$
  - (semantic notions are fully covered by our syntactic rules)
  - (Means: we wrote down *enough* rules!)

- We will prove the contrapositive, using induction:
- **Complete**: If $\Gamma \nvdash_{STD} \Theta$, then $\Gamma \nvDash \Theta$
- "if an argument is not tree-valid, then it's not semantically valid"

Fleshing out the Proof Idea

- "Not tree-valid" means that EVERY tree with root $\Gamma \cup \{\sim\Theta\}$ remains **open** (i.e. never closes, even after resolving every sentence)
- Call this open branch '**Oprah**' (or 'O' for short)
- Use Oprah to define a TVA '$\mathcal{I}$' that makes true every wff on O
- So in particular, $\mathcal{I}$ will make true the sentences in the root, showing they are **consistent** $\Rightarrow$ argument is semantically invalid: i.e. $\mathcal{I}$ makes Γ true and $\sim\Theta$ true, i.e. Θ FALSE
- This will prove: **Completeness**: If $\Gamma \nvdash_{STD} \Theta$, then $\Gamma \nvDash \Theta$

Starting the Completeness Proof!

- Consider an arbitrary argument with premise set Γ and putative conclusion Θ
- Construct a tree with root $\Gamma \cup \{\sim\Theta\}$
- Assume that this argument is NOT tree valid: so every such tree is open and extends to a complete open tree
- Hence, there must be at least one complete open branch: O
- Notice that O must end with atomic sentences or negations of these in its last node
- (So we see that we could do induction, starting 'from the bottom' of our tree)

Setting up the Induction (HW question!)

- At this point, we would need to (i) decide what we want to do induction over
- (ii) Handle the base case(s)
- (iii) State the induction hypothesis
- (iv) Proceed to handle all cases in the induction step (coming from a recursive definition(s))
- We’ll leave this to an optional HW question and proceed casually

Using Oprah to define $\mathcal{I}$

- Since Oprah is open, no wff and its negation ever appear on it

- Hence, we can define a TVA '$\mathcal{I}$' as follows:

  - $\mathcal{I}$ assigns 'False' to an atomic sentence if and only if its negation appears by itself on a node of O

  - Hence, $\mathcal{I}$ assigns 'True' to every atomic sentence appearing by itself on one of O's nodes

Idea: show $\mathcal{I}$ makes true every wff on O

- Consider an arbitrary wff Δ on Oprah
- If Δ is an atomic sentence or negated atomic sentence, then $\mathcal{I}$ makes it true by definition
- Otherwise, there must exist wff α and β that compose Δ according to our recursive definition of SL sentences
- Since Oprah is complete, Δ must be resolved according to one of our nine tree rules, leading to sentences on the ‘child nodes’, and one of these child nodes must lie on Oprah
- By induction, we assume that the child node on O is satisfied by $\mathcal{I}$
- Given this, we aim to show that $\mathcal{I}$ makes Δ true, **no matter which** child node lies on Oprah and is satisfied by $\mathcal{I}$

What we can conclude AFTER the subcases:

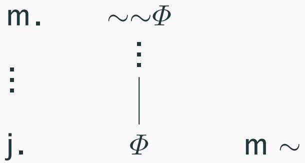
- Assuming we complete our heroic quest, we get to conclude that $\mathcal{I}$ makes Δ true
- But Δ was an arbitrary wff on Oprah
- So we'll have shown that $\mathcal{I}$ makes each sentence in the root true
- By showing that the root is **satisfiable**, $\mathcal{I}$ shows that the argument is **semantically invalid**, which is what we wanted to show!
- We'll have used Oprah to define a TVA that makes the premises true but the conclusion false

Reasoning from the 'Bottom Up'

- Notice that we are reasoning from the sentences *below* the resolved sentence, back up the tree
- We need to show that *for **EACH** child node*, if $\mathcal{I}$ satisfies it, then $\mathcal{I}$ makes the resolved sentence Δ true
- Effectively, we are showing that each child node separately entails the resolved sentence
- There are nine cases to consider, coming from our nine tree rules

Subcase i) Δ is resolved by Double Negation ($\sim$)

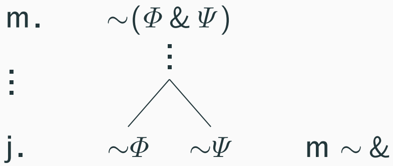
Double Negation ($\sim$)



- Only one child node $\Rightarrow$ Φ lies on O
- By induction hypothesis, $\mathcal{I}$ makes Φ true
- Hence, $\mathcal{I}$ must assign true to $\sim\sim\Phi$, which in this case is Δ
- So $\mathcal{I}$ satisfies Δ
- On to the next one!

Subcase iii) Δ is resolved by Negated Conjunction ($\sim\&$)

Negated Conjunction ($\sim\&$)



- Exactly one child node lies on O, but we don't know which one (so we must show $\mathcal{I}$ satisfies Δ either way!)
- a) If $\sim\Phi$ lies on O, then $\mathcal{I}$ makes $\sim\Phi$ true (by induction hypothesis)
  $\Rightarrow \mathcal{I}$ makes Φ false
  $\Rightarrow \mathcal{I}$ makes $(\Phi \& \Psi)$ false, and hence makes Δ true
- b) Likewise if $\sim\Psi$ lies on O...
- Either way, $\mathcal{I}$ satisfies Δ

Subcase v) Δ is resolved by Negated Disjunction ($\sim\vee$)

Negated Disjunction ($\sim\vee$)

m.	$\sim(\Phi \vee \Psi)$	
⋮	⋮	
j.	$\sim\Phi$	m $\sim\vee$
j+1.	$\sim\Psi$	

- Only one child node $\Rightarrow$ both $\sim\Phi$ and $\sim\Psi$ lie on O
- By induction hypothesis, $\mathcal{I}$ makes both children wff true
- Hence, $\mathcal{I}$ must make false both Φ and Ψ
  $\Rightarrow \mathcal{I}$ makes false $(\Phi \vee \Psi)$
  $\Rightarrow \mathcal{I}$ makes true $\sim(\Phi \vee \Psi)$
- So $\mathcal{I}$ satisfies Δ

A Key fact about Complete Open Trees

- A complete open tree has at least one complete open branch
- A complete open branch never contains both a sentence Φ and its negation $\sim\Phi$
- In a 'partially complete system', **our construction above lets us define a TVA '$\mathcal{I}$' that makes true each sentence on a complete open branch**, including the root
- Upshot: if an argument is tree-invalid (i.e. has at least one complete open tree) in a 'partially complete' system , then there is a truth-value assignment that satisfies the root (so $\Gamma \nvDash \Theta$)
- If the system is also SOUND, one could then conclude that the argument is *not* tree-valid.

What are the ‘partially complete’ systems?

- Our construction of a TVA that satisfies each wff on the complete open branch just requires that *for the rules we use*, the sentences on each child node entail the sentence being resolved.
- Call systems “partially complete” if they have this property
- Note that the system STD^{conj} which has only the rules ($\sim$), (&), and ($\sim$ &) is partially complete in this sense:
- for the sentences that *can be* completely resolved with these limited rules, we can turn any complete open branch into a TVA that satisfies all wff on the branch.

5. Metalogic for STD

e. Testing Alternative Rules: Completeness

Checking Completeness: “bottom-up” reasoning

- **Heuristic for Completeness checking**: ask whether **EACH** child node (below) individually entails the sentence being resolved (i.e. the sentence ‘up top’, in the ‘parent-node’)
- If ‘**yes**’, then the rule preserves completeness: proceed to extend our completeness proof for this case (reasoning in terms of arbitrary TVA’s, one for each child node).
- If ‘**no**’, then the rule BREAKS completeness: proceed to **construct a counter-example** using the rule.

Counterexamples to Completeness

- To show completeness fails, it is NECESSARY to provide a CONCRETE counterexample, using SL sentences
  - Further heuristic reasoning about TVAs is not enough!

- What you need for a counterexample to completeness:
  1.) Choose an **UNsatisfiable root**, i.e. an INconsistent set of sentences
  2.) Apply the modified rule in the tree; you may also use other rules that we've already shown preserve completeness
  3.) Show that the tree has a **complete open branch**, i.e. does NOT close

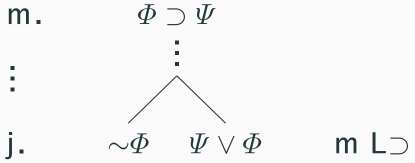
Why such Counterexamples work

- **Complete**: If $\Gamma \vDash \Theta$, then $\Gamma \vdash_{STD*} \Theta$
- Counterexample: $\Gamma \vDash \Theta$ but $\Gamma \nvdash_{STD*} \Theta$
  - '$\Gamma \vDash \Theta$' means that $\Gamma \cup \{\sim\Theta\}$ is INconsistent (i.e. UNsatisfiable) (i.e. any TVA that makes Γ true makes the conclusion Θ true)
  - '$\Gamma \nvdash_{STD*} \Theta$' means that it is NOT the case that the argument is tree–valid in $STD*$ (a claim about ALL trees)
  - From completeness proof, we know that if the system *were* complete, then we could use any **complete open branch** to construct a TVA that **satisfies** the root
  - Hence, if we construct a **complete open tree** with an **unsatisfiable root**, this is a reductio of completeness

Liberal Conditional and Conservative Biconditional

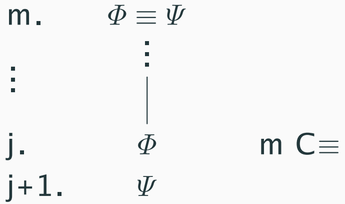
*STD**: replace rule (⊃) with:

Liberal Conditional (L⊃)



STD†: replace rule (≡) with:

Conservative Biconditional (C≡)

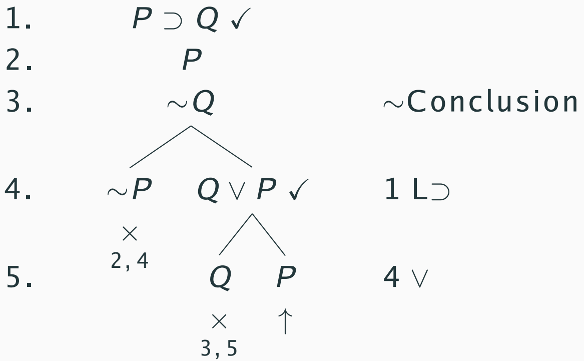


Liberal Conditional (now from the bottom up!)

- Reason from the bottom–up, handling each child node separately
- $\sim\Phi$ entails $\Phi \supset \Psi$
- $(\Psi \vee \Phi)$ does NOT entail $\Phi \supset \Psi$, since Φ doesn't entail it
- (Remember: $\Phi \supset \Psi$ is equivalent to $\sim\Phi \vee \Psi$)
- Hence, we **proceed to counterexample!**

Counterexample to Completeness of STD^*

- Counterexample: find complete open tree with unsatisfiable root:



- Note that $\{P \supset Q,\ P\} \vDash Q$ but **tree-invalid** in STD^*
- If STD^* *were* complete, the complete open branch would lead to a TVA that satisfies the root. Contradiction $\Rightarrow$ not complete

A Common Mistake to Avoid!!!!

- You are very liable to forget to COMPLETE YOUR open branch!!!
- You only have a counterexample to completeness if you have a COMPLETE open branch with an unsatisfiable root
- So make sure you FULLY RESOLVE every sentence on the open branch, until you can write that coveted up arrow ‘↑’!
- When in doubt, just complete the whole tree

Conservative Biconditional (bottoms up!)

- Reason from the bottom up: there is only one child node. So we only need its sentences to entail $\Phi \equiv \Psi$
- Sentences on bottom are equivalent to $(\Phi \,\&\, \Psi)$, which DOES entail $\Phi \equiv \Psi$. So we proceed to formally extend our completeness proof:
    - **Formally**: since only one child node, both Φ and Ψ lie on the complete open branch O.
    - By induction hypothesis, both are true according to $\mathcal{I}$.
    - Hence, $\Phi \equiv \Psi$ is true on $\mathcal{I}$, which is what we needed to show.

Some Fun Questions to Ponder

- What is the least number of tree rules required for soundness? For completeness?
    - We'll return to this question if we ever get around to discussing the "expressive adequacy" of a given set of connectives

- When assessing how a modification affects soundness, do we need to know anything about the other rules?

- When assessing how a modification affects completeness, do we need to know anything about the other rules?

6. Proofs in SL

6. Proofs in SL

a. Who ordered *that*???

The Power of a Proof System

- We have already seen how powerful a proof system can be compared to truth tables:
- Consider our running example argument:

$$C \vee E, A \vee M, A \supset {\sim}C, {\sim}S \,\&\, {\sim}M \therefore E$$

  requires 32 lines and 512 truth values

- With trees, we showed validity in 10 lines (13 sentences)
- Why not just tree always and everywhere?

For the Love of Trees

- The advantages of trees create limitations:
  - Mechanical $\Rightarrow$ not ‘normal’ pattern of inference
  - Mirror partial truth-tables $\Rightarrow$ implicitly referencing truth-values
  - Always checking satisfiability of the root $\Rightarrow$ we don’t really ‘derive’ anything at the end of the proof, unlike in normal inference
  - (this is why so many of us keep forgetting that it is the NEGATION of the conclusion that goes in the root)
- We would like to form a better model of human inference patterns
- e.g., common rules such as Modus Ponens, disjunctive syllogism

Idiosyncrasies of Table or Tree Reasoning

- Tables:
  - Construct a truth table
  - verify there is no TVA where premises are true but conclusion is false
- This is NOT how we typically reason through an argument
- Trees: ask whether a set of sentences is satisfiable:
  - Put premises and NEGATION of conclusion in root
  - If tree closes, then unsatisfiable root (valid argument)
  - If tree remains open, then satisfiable root (invalid argument)
- Again, this is NOT how we typically reason through an argument

The Very Idea of ‘Natural Deduction’

- We commonly reason according to certain inference rules
- And we often make assumptions in the *middle* of our reasoning, derive an intermediate conclusion, and ‘discharge’ the assumption (e.g. in proof by contradiction)
- We would like to see if we can *vindicate* these patterns:
  - Show that these rules never get us into trouble: soundness
  - Show that we have enough rules to handle any valid argument (including additional rules we might want to add): completeness
- Perhaps our natural deduction system *explains* the success of our ordinary inference patterns

Ordinary Reasoning by “Proofs”

- Idea: work our way from premises to conclusion using steps we know are entailed by the premises.
- For instance:
  - From “Neither Sarah nor Amir enjoys hiking” we can conclude “Amir doesn’t enjoy hiking.”
  - From “Either Amir lives in Chicago or he enjoys hiking” and “Amir doesn’t enjoy hiking” we can conclude “Amir lives in Chicago” (Disjunctive syllogism DS).
  - etc.
- If we manage to work from the premises to the conclusion in this way, we know that the argument must be valid.

An informal proof

Our argument

1. Sarah lives in Chicago or Erie.
2. Amir lives in Chicago unless he enjoys hiking.
3. If Amir lives in Chicago, Sarah doesn't.
4. Neither Sarah nor Amir enjoy hiking.

$\therefore$ Sarah lives in Erie.

5. Amir doesn't enjoy hiking (from 4).
6. Amir lives in Chicago (from 2 and 5).
7. Sarah doesn't live in Chicago (from 3 and 6).
8. Sarah lives in Erie (from 1 and 7).

A more formal proof

Our argument

1. $C \vee E$
2. $A \vee M$
3. $A \supset {\sim}C$
4. ${\sim}S \,\&\, {\sim}M$

$\therefore E$

5. ${\sim}M$ (from 4, since $\mathcal{P} \,\&\, \mathcal{Q} \vDash \mathcal{Q}$)
6. A (from 2 and 5, since $\{\mathcal{P} \vee \mathcal{Q}, {\sim}\mathcal{Q}\} \vDash \mathcal{P}$)
7. ${\sim}C$ (from 3 and 6, since $\{\mathcal{P} \supset \mathcal{Q}, \mathcal{P}\} \vDash \mathcal{Q}$, i.e. via ‘modus ponens’)
8. E (from 1 and 7, since $\{\mathcal{P} \vee \mathcal{Q}, {\sim}\mathcal{P}\} \vDash \mathcal{Q}$)

Some aspects of our Formal Deductions

- Numbered lines contain sentences of SL
- A line may be a **premise** (:PR).
- A line may be an **assumption** (:AS)
- If neither a premise nor assumption, it must be **justified**
- Justification requires:
  - a **rule** (e.g. ‘&E’), and
  - prior line(s) invoked by the rule—referenced by line number(s)
  - starting with a colon: e.g. ‘: 2 &E’
- But: what are the rules? (very different from ‘what IS a rule’?)

Aspects of our Rules for Natural Deduction

- Our Rules will (mostly) be ...
  - **Simple**: cite just a few lines as justification
  - **Obvious**: new line should clearly be entailed by justifications
  - **Schematic**: can be described just by **forms** of sentences involved
  - **Few in number**: want to make do with just a handful
- We'll have two rules per connective:
  an **introduction** and an **elimination** rule
- They'll be used to either:
  - justify (say) $\mathcal{P} \,\&\, \mathcal{Q}$ (i.e. to 'introduce' $\&$), or
  - justify something **using** $\mathcal{P} \,\&\, \mathcal{Q}$ (i.e. to 'eliminate' $\&$).

6. Proofs in SL

b. Conjunction Intro and Elimination (Rules for $\&$)

Eliminating &

- What can we **justify using** $\mathcal{P} \,\&\, \mathcal{Q}$?
- A conjunction entails each conjunct:

$$\mathcal{P} \,\&\, \mathcal{Q} \vDash \mathcal{P}$$
$$\mathcal{P} \,\&\, \mathcal{Q} \vDash \mathcal{Q}$$

- Already used this above to get $\sim M$ from $\sim S \,\&\, \sim M$, i.e., from "Neither Sarah nor Amir enjoys hiking" we concluded "Amir doesn't enjoy hiking".
- (Role of $\mathcal{P}$ played by $\sim S$ and that of $\mathcal{Q}$ played by $\sim M$)

Introducing &

- What do we **need to justify** $\mathcal{P} \,\&\, \mathcal{Q}$?
- We need both $\mathcal{P}$ and $\mathcal{Q}$:

$$\{\mathcal{P}, \mathcal{Q}\} \vDash \mathcal{P} \,\&\, \mathcal{Q}$$

- For instance, if we have “Sarah doesn’t enjoy hiking” and also “Amir doesn’t enjoy hiking”, we can conclude “Neither Sarah nor Amir enjoys hiking”
- (Role of $\mathcal{P}$ played by $\sim S$ and $\mathcal{Q}$ by $\sim M$: $\{\sim S, \sim M\} \vDash \sim S \,\&\, \sim M$)

Rules for &

$$\begin{array}{r|ll} m & \mathcal{P} & \\ n & \mathcal{Q} & \\ & \mathcal{P} \,\&\, \mathcal{Q} & :m,\ n\ \ \&\mathrm{I} \end{array}$$

$$\begin{array}{r|ll} m & \mathcal{P} \,\&\, \mathcal{Q} & \\ & \mathcal{P} & :m\ \ \&\mathrm{E} \end{array}$$

$$\begin{array}{r|ll} m & \mathcal{P} \,\&\, \mathcal{Q} & \\ & \mathcal{Q} & :m\ \ \&\mathrm{E} \end{array}$$

We’ll illustrate using exercises in our Week 6 Practice Problems on Carnap.

$$\begin{array}{r|ll}
1 & A \mathbin{\&} B & :\mathrm{PR} \\ \cline{2-2}
2 & A & :1\ \ \&\mathrm{E} \\
3 & B & :1\ \ \&\mathrm{E} \\
4 & B \mathbin{\&} A & :2,\ 3\ \ \&\mathrm{I}
\end{array}$$

1	$A \& (B \& C)$	:PR
2	A	:1 & E
3	$B \& C$	:1 & E
4	C	:3 & E
5	$A \& C$	:2, 4 & I

6. Proofs in SL

c. Conditional Intro and Elim. (Rules for ⊃)

Eliminating $\supset$

- What can we **justify using** $\mathcal{P} \supset \mathcal{Q}$?
- We used the conditional "If Amir lives in Chicago, Sarah doesn't" to justify "Sarah doesn't live in Chicago".
- What is the general rule? What can we justify using $\mathcal{P} \supset \mathcal{Q}$? What do we need in addition to $\mathcal{P} \supset \mathcal{Q}$?
- The principle is **modus ponens** (affirming the antecedent):

$$\{\mathcal{P} \supset \mathcal{Q}, \mathcal{P}\} \vDash \mathcal{Q}$$

- (When inferring from $A \supset {\sim}C$ and A to ${\sim}C$, the role of $\mathcal{P}$ is played by A and role of $\mathcal{Q}$ by ${\sim}C$.)

Elimination rule for ⊃

$$
\begin{array}{r|ll}
m & \mathcal{P} \supset \mathcal{Q} & \\
n & \mathcal{P} & \\
 & \mathcal{Q} & \quad : m,\ n \supset\mathrm{E}
\end{array}
$$

Let’s illustrate this rule using an exercise in Carnap:
we show that $\{A \,\&\, B, A \supset C, B \supset D\} \vDash C \,\&\, D$.

1	$A \& B$	:PR
2	$A \supset C$	:PR
3	$B \supset D$	:PR
4	A	:1 & E
5	C	:2, 4 ⊃E
6	B	:1 & E
7	D	:3, 6 ⊃E
8	$C \& D$	:5, 7 & I

Introducing $\supset$

- How do we justify a conditional? What should we require for a proof of $\mathcal{P} \supset \mathcal{Q}$ (say, from some premise $\mathcal{R}$)?
- We need a proof that shows that $\mathcal{R} \vDash \mathcal{P} \supset \mathcal{Q}$.
- Idea: show instead that $\mathcal{R}, \mathcal{P} \vDash \mathcal{Q}$.
- The conditional $\supset$ no longer appears, so this seems easier.
- It's a good move, because if $\mathcal{R}, \mathcal{P} \vDash \mathcal{Q}$ then $\mathcal{R} \vDash \mathcal{P} \supset \mathcal{Q}$.

Justifying $\supset$I

Fact

If $\mathcal{R}, \mathcal{P} \vDash \mathcal{Q}$ then $\mathcal{R} \vDash \mathcal{P} \supset \mathcal{Q}$.

- If $\mathcal{R}, \mathcal{P} \vDash \mathcal{Q}$ then every TVA makes $\mathcal{R}$ or $\mathcal{P}$ false or it makes $\mathcal{Q}$ true
- Let's show that no valuation is a counterexample to $\mathcal{R} \vDash \mathcal{P} \supset \mathcal{Q}$:
  1. A valuation that makes $\mathcal{R}$ and $\mathcal{P}$ true, but $\mathcal{Q}$ false, is impossible if $\mathcal{R}, \mathcal{P} \vDash \mathcal{Q}$.
  2. So any valuation must make $\mathcal{R}$ false, $\mathcal{P}$ false, or $\mathcal{Q}$ true.
  3. If it makes $\mathcal{R}$ false, it's not a counterexample to $\mathcal{R} \vDash \mathcal{P} \supset \mathcal{Q}$.
  4. If it makes $\mathcal{P}$ false, it makes $\mathcal{P} \supset \mathcal{Q}$ true, so it's not a counterexample.
  5. If it makes $\mathcal{Q}$ true, it also makes $\mathcal{P} \supset \mathcal{Q}$ true, so it's not a counterexample.
- So, there are no counterexamples to $\mathcal{R} \vDash \mathcal{P} \supset \mathcal{Q}$.
- If $\mathcal{R} = \varnothing$, then from $\mathcal{P} \vDash \mathcal{Q}$ we can infer $\vDash \mathcal{P} \supset \mathcal{Q}$

Subproofs (CRUCIAL CONCEPT)

- We want to justify $\mathcal{P} \supset \mathcal{Q}$ by giving a proof of $\mathcal{Q}$ from *assumption* $\mathcal{P}$ (and possibly other premises Γ, e.g. $\mathcal{R}$)
- How to do this in a proof? We can add something as a premise and *discharge* it later!
- Solution: add $\mathcal{P}$ as an assumption (:AS), and keep track of what depends on that assumption (say, by indenting and a vertical line)
- Once we're done (have proved $\mathcal{Q}$), close this "subproof".
- Justification of $\mathcal{P} \supset \mathcal{Q}$ is the **entire** subproof (use a HYPHEN)
- **Important**: nothing **inside** a subproof is available outside as a justification (since inner lines might depend on the assumption)

Introduction rule for ⊃

$$
\begin{array}{l|l|ll}
m & & \mathcal{P} & :\text{AS for } \supset\text{I} \\ \cline{3-3}
 & & \vdots & \\
n & & \mathcal{Q} & \\
 & \multicolumn{2}{l}{\mathcal{P} \supset \mathcal{Q}} & : m\text{-}n \supset\text{I}
\end{array}
$$

NOTE THE **HYPHEN** IN THE JUSTIFICATION LINE!!!

We'll illustrate using more exercises from Week 6 Practice Problems

- Show: $\{A \supset B, B \supset C\} \vDash A \supset C$.
- Show: $A \supset (B \supset C) \vDash (A \,\&\, B) \supset (A \,\&\, C)$

```
1 | A ⊃ B        :PR
2 | B ⊃ C        :PR
  |--------
3 |   | A        :AS for ⊃I
  |   |----
4 |   | B        :1, 3 ⊃E
5 |   | C        :2, 4 ⊃E
6 | A ⊃ C        :3-5 ⊃I
```

1	$A \supset (B \supset C)$	:PR
2	$\mid\ A \& B$	:AS for ⊃I
3	$\mid\ A$	:2 & E
4	$\mid\ B \supset C$	:1, 3 ⊃E
5	$\mid\ B$	:2 & E
6	$\mid\ C$	:4, 5 ⊃E
7	$\mid\ A \& C$	:3, 6 & I
8	$(A \& B) \supset (A \& C)$	:2-7 ⊃I

6. Proofs in SL

d. Use of subproofs

Reiteration (for the 11th hour!)

$\mathcal{P} \vDash \mathcal{P}$, so “Reiteration” R is a good rule:

$$\begin{array}{r|ll} m & \mathcal{P} & \\ & & \\ k & \mathcal{P} & : m\ \mathrm{R} \end{array}$$

Uses of reiteration (to the Carnap!):

- Proof of $A \vDash A$.
- Proof that $A \supset (B \supset A)$ is a tautology.

1	$\quad A$	:AS for ⊃I
2	$\quad A$	:1 R
3	$A \supset A$	:1-2 ⊃I

Again, note the **HYPHEN**! Even though our subproof is only two lines, we still write ‘:1–2’ and NOT ‘:1, 2’.

```
1 │ │ A                 :AS for ⊃I
  │ ├────
2 │ │ │ B               :AS for ⊃I
  │ │ ├────
3 │ │ │ A               :1 R
4 │ │ B ⊃ A             :2-3 ⊃I
5 │ A ⊃ (B ⊃ A)         :1-4 ⊃I
```

Rules for justifications and subproofs

- When a rule calls for a subproof, we cite it as ": *m*–*n*", hyphenating the first and last line numbers of the subproof.
- Sentences on the Assumption line **and** last line MUST match rule
- After a subproof is done, you can only cite the whole thing, NOT any line in it (you are 'outside the scope' of these lines)
- Subproofs (subproofs can be nested) can be nested
- You also can't cite any subproof entirely contained inside another subproof, once the surrounding subproof is completed (since again you'd be 'outside the scope' of those lines)

Reiteration (do’s a don’ts) DOs and DON’Ts!

Which are correct applications of R?

```
1 │ │ A       :AS
  │ ├──
2 │ │ │ A     :AS
  │ │ ├──
3 │ │ │ A     :1 ✓ R
4 │ │ A       :1 ✓ R
5 │ │ A       :2 ✗ R
6 │ A         :2 ✗ R
7 │ A         :1 ✗ R
```

6. Proofs in SL

e. Disjunction Intro and Elim. (Rules for $\vee$)

Introduction rule for $\lor$

We have $\mathcal{P} \vDash \mathcal{P} \lor \mathcal{Q}$. So:

$$\begin{array}{r|ll} m & \mathcal{P} & \\ & \mathcal{P} \lor \mathcal{Q} & :m \lor\mathrm{I} \end{array} \qquad\qquad \begin{array}{r|ll} m & \mathcal{Q} & \\ & \mathcal{P} \lor \mathcal{Q} & :m \lor\mathrm{I} \end{array}$$

- Note that the introduced disjunct can be ANYTHING!
- And you can introduce on the left OR right side!
- Let's do practice problem 6.10 on Carnap!

1	$\quad \mid A$	:AS for $\supset$I
2	$\quad \mid B \vee A$	:1 $\vee$I
3	$A \supset (B \vee A)$	:1–2 $\supset$I

Eliminating $\vee$ (Proof by Cases)

- What can we justify with disjunction $\mathcal{P} \vee \mathcal{Q}$?
- Not $\mathcal{P}$ and also not $\mathcal{Q}$: neither is entailed by $\mathcal{P} \vee \mathcal{Q}$.
- But: if both $\mathcal{P}$ and $\mathcal{Q}$ separately entail some third sentence $\mathcal{R}$, then we know that $\mathcal{R}$ follows from the disjunction!
- To show this, we need **two** subproofs that show $\mathcal{R}$, but in each proof we are allowed to use only one of $\mathcal{P}$, $\mathcal{Q}$.

Elimination rule for $\vee$ (Proof by Cases)

```
m  | $\mathcal{P} \vee \mathcal{Q}$
i  | | $\mathcal{P}$                    :AS for $\vee$E
   | |----
   | | $\vdots$
j  | | $\mathcal{R}$
   | ––
k  | | $\mathcal{Q}$                    :AS for $\vee$E
   | |----
   | | $\vdots$
ℓ  | | $\mathcal{R}$
   | $\mathcal{R}$                      :$m$, $i$–$j$, $k$–$\ell$ $\vee$E
```

- From $\mathcal{P}$ we derive $\mathcal{R}$
- Start a subproof for each disjunct
- The subproofs need not be adjacent, but if they are, **separate with ––**
- From $\mathcal{Q}$ we derive $\mathcal{R}$
- You can swap the order of the subproofs
- Remember to cite BOTH subproofs (hyphens!), AND the line with the disjunction
- Remember to pop out of subproof level at the end!

```
1 | A ∨ B        :PR
  |-------
2 | | A          :AS for ∨E
  | |----
3 | | B ∨ A      :2 ∨I
  |
4 | | B          :AS for ∨E
  | |----
5 | | B ∨ A      :4 ∨I
6 | B ∨ A        :1, 2-3, 4-5 ∨E
```

- In Carnap: Need -- between the subproofs
- Note: need the **SAME sentence** as the last line of each subproof
- Note the complex justification structure: (a) line with disjunction, (b) first subproof, (c) second subproof, (d) the rule itself
- Proceed to Carnap PP6.15!

1	$A \vee B$	:PR
2	$\underline{A \supset B}$	:PR
3	$\quad \vert\ \underline{A}$	:AS for $\vee$E
4	$\quad \vert\ B$	:2, 3 $\supset$E
5	$\quad \vert\ \underline{B}$	:AS for $\vee$E
6	$\quad \vert\ B$	:5 R
7	B	:1, 3-4, 5-6 $\vee$E

6. Proofs in SL

f. Negation Intro and Elimination

Introducing $\sim$

- An argument is valid iff the premises together with the negation of the conclusion are jointly unsatisfiable.
- For instance:
  - $\mathcal{Q} \vDash \mathcal{P}$ iff $\mathcal{Q}$ and $\sim\mathcal{P}$ are jointly unsatisfiable.
  - $\mathcal{Q} \vDash \sim\mathcal{P}$ iff $\mathcal{Q}$ and $\mathcal{P}$ are jointly unsatisfiable.
- This last one gives us idea for $\sim$I rule: To justify $\sim\mathcal{P}$, show that $\mathcal{P}$ (together with all other premises) is unsatisfiable.
- Unsatisfiable means: a contradiction follows!

Negation Introduction ($\sim$I)

m	$\quad\vert\ \Phi$	:AS for $\sim$I
	$\quad\vert\ \vdots$	
n	$\quad\vert\ \Psi$	
	$\quad\vert\ \vdots$	
o	$\quad\vert\ \sim\Psi$	
	$\sim\Phi$	:*m*-*o* $\sim$I

- Assume the **non**–negated wff!
- Derive a sentence and its negation (could be Φ!)
- (Ψ and $\sim\Psi$ can appear in opposite order)
- Pop out of the subproof and **introduce** that negativity!
- Remember to cite the WHOLE subproof (hyphen!)
- Let’s try exercise PP6.21:

```
1 │ │ A ⊃ B                        :AS for ⊃I
  │ ├──────
2 │ │ │ ∼B                         :AS for ⊃I
  │ │ ├─────
3 │ │ │ │ A                        :AS for ∼I
  │ │ │ ├────
4 │ │ │ │ B                        :1, 3 ⊃E
5 │ │ │ │ ∼B                       :2 R
6 │ │ │ ∼A                         :3-5 ∼I
7 │ │ ∼B ⊃ ∼A                      :2-6 ⊃I
8 │ (A ⊃ B) ⊃ (∼B ⊃ ∼A)            :1-7 ⊃I
```

Negation Elimination ($\sim$E)

m	$\sim\Phi$	:AS for $\sim$E
	$\vdots$	
n	Ψ	
	$\vdots$	
o	$\sim\Psi$	
	Φ	:m-o $\sim$E

- Assume the **negated** wff!
- Derive a sentence and its negation (could be Φ!)
- (Ψ and $\sim\Psi$ can appear in opposite order)
- Pop out of the subproof and **eliminate** that negativity!
- Put a smile on!
- Remember to cite the WHOLE subproof (hyphen!)
- Let's try exercise PP6.22:

```
1 | | ~A ⊃ ~B              :AS for ⊃I
  | |----------
2 | | | B                  :AS for ⊃I
  | | |----
3 | | | | ~A             :AS for ~E
  | | | |-----
4 | | | | ~B             :1, 3 ⊃E
5 | | | | B              :2 R
6 | | | A                  :3-5 ~E
7 | | B ⊃ A                :2-6 ⊃I
8 | (~A ⊃ ~B) ⊃ (B ⊃ A)    :1-7 ⊃I
```

6.f.5

6. Proofs in SL

g. Biconditional Intro and Elimination (<->)

Biconditional Introduction ($\equiv$I) (Type <-> !!!)

i	$\mathcal{A}$	:AS for $\equiv$I
	$\vdots$	
j	$\mathcal{B}$	
	--	
k	$\mathcal{B}$	:AS for $\equiv$I
	$\vdots$	
l	$\mathcal{A}$	
	$\mathcal{A} \equiv \mathcal{B}$	:i-j, k-l $\equiv$I

- Like doing conditional intro twice, from both directions
- You can swap the order of the subproofs
- The subproofs need not be adjacent, but if they are, **separate with --**
- Remember to cite BOTH subproofs (hyphens!)
- Remember to pop out of subproof line!

Biconditional Elimination ($\equiv$E) (Type <-> !!!)

$$\begin{array}{r|ll} m & \mathcal{A} \equiv \mathcal{B} & \\ n & \mathcal{A} & \\ & \mathcal{B} & :m,\ n \equiv\text{E} \end{array}$$

$$\begin{array}{r|ll} m & \mathcal{A} \equiv \mathcal{B} & \\ n & \mathcal{B} & \\ & \mathcal{A} & :m,\ n \equiv\text{E} \end{array}$$

- Just like conditional elimination!
- Only now you can eliminate from either side! (power!)
- There can be lines between lines m and n
- Remember to cite the lines of both (i) the biconditional and (ii) the side you have already
- Carnap issue: must type <->E

Issue with Typing $\equiv$ in Carnap

- For Carnap to recognize $\equiv$I or $\equiv$E in the justification column, you sadly must type $<->$ I or $<->$ E
- This is a bummer; I hope we can have it fixed (eventually)
- It is still fine to type $<>$ for the biconditional symbol in the sentences
- You can also copy/paste the $\equiv$symbol from elsewhere on the page!

6. Proofs in SL

h. Strategies and examples

Working forward and backward

- **Working backward** from a conclusion (goal) means:
  - Find main connective of goal sentence
  - Match with conclusion of corresponding **I**ntro rule
  - Write out (above the goal!) what you'd need to apply that rule
- **Working forward** from a premise, assumption, or already justified sentence means:
  - Find main connective of premise, assumption, or sentence
  - Match with top premise of corresponding E rule
  - Write out what else you need to apply the E rule (new goals)
  - If necessary, write out conclusion of the rule

Constructing a proof

- Write out premises at the top (if there are any)
- Write conclusion at bottom
- Work backward & forward from goals and premises/assumptions in this order:
    - Work backward using $\&$I, $\supset$I, $\equiv$I, $\sim$I/E, or forward using $\vee$E
    - Work forward using $\&$E
    - Work forward using $\supset$E, $\equiv$E
    - Work backward from $\vee$I
    - Try Negation Intro or Elimination, working toward a contradiction
- Repeat for each new goal from top

6. Proofs in SL

i. The Rules, Reiterated

The rules, one more time: Reiteration

$$
\begin{array}{r|ll}
m & \mathcal{P} & \\
 & \vdots & \\
k & \mathcal{P} & :m\ \mathrm{R}
\end{array}
$$

- Remember that you must be in the scope of the line you're reiterating
- e.g. if you're outside a subproof, you can't reiterate anything wholly within the subproof

The rules: Conjunction Intro (&I) and Elimination (&E)

$$
\begin{array}{r|ll}
m & \mathcal{P} & \\
n & \mathcal{Q} & \\
 & \mathcal{P} \,\&\, \mathcal{Q} & :m,\ n\ \&\mathrm{I}
\end{array}
$$

$$
\begin{array}{r|ll}
m & \mathcal{P} \,\&\, \mathcal{Q} & \\
 & \mathcal{P} & :m\ \&\mathrm{E}
\end{array}
$$

$$
\begin{array}{r|ll}
m & \mathcal{P} \,\&\, \mathcal{Q} & \\
 & \mathcal{Q} & :m\ \&\mathrm{E}
\end{array}
$$

The rules: Conditional Intro (⊃I) and Elim (⊃E)

$$\begin{array}{l|l|ll}
m & & \mathcal{P} & :\text{AS for } \supset\text{I} \\
\cline{3-3}
 & & \vdots & \\
n & & \mathcal{Q} & \\
 & \multicolumn{2}{l}{\mathcal{P} \supset \mathcal{Q}} & :m\text{–}n \supset\text{I}
\end{array}$$

$$\begin{array}{l|ll}
m & \mathcal{P} \supset \mathcal{Q} & \\
n & \mathcal{P} & \\
 & \mathcal{Q} & :m,\ n \supset\text{E}
\end{array}$$

The rules: Disjunction Intro (∨I) and Elimination (∨E)

$$
\begin{array}{r|l l}
m & \mathcal{P} \vee \mathcal{Q} & \\
i & \quad \begin{array}{|l} \mathcal{P} \\ \hline \end{array} & :\text{AS for } \vee\text{E} \\
 & \quad \begin{array}{|l} \vdots \end{array} & \\
j & \quad \begin{array}{|l} \mathcal{R} \end{array} & \\
 & -- & \\
k & \quad \begin{array}{|l} \mathcal{Q} \\ \hline \end{array} & :\text{AS for } \vee\text{E} \\
 & \quad \begin{array}{|l} \vdots \end{array} & \\
\ell & \quad \begin{array}{|l} \mathcal{R} \end{array} & \\
 & \mathcal{R} & :m,\ i\text{–}j,\ k\text{–}\ell\ \vee\text{E}
\end{array}
$$

$$
\begin{array}{r|l l}
m & \mathcal{P} & \\
 & \mathcal{P} \vee \mathcal{Q} & :m\ \vee\text{I}
\end{array}
$$

$$
\begin{array}{r|l l}
m & \mathcal{Q} & \\
 & \mathcal{P} \vee \mathcal{Q} & :m\ \vee\text{I}
\end{array}
$$

Remember that $\mathcal{P}$ can be the same wff as $\mathcal{Q}$

so can introduce $\mathcal{P} \vee \mathcal{P}$ from $\mathcal{P}$

The rules: Negation Intro and Elimination

Negation Intro (∼I)

```
m  | | Φ        :AS for ∼I
   | |----
   | | ⋮
n  | | Ψ
   | | ⋮
o  | | ∼Ψ
   | ∼Φ         :m-o ∼I
```

Neg. Elimination (∼E)

```
m  | | ∼Φ       :AS for ∼E
   | |-----
   | | ⋮
n  | | Ψ
   | | ⋮
o  | | ∼Ψ
   | Φ          :m-o ∼E
```

Note that you can swap the order of Ψ and $\sim\Psi$ in the subproofs!

The rules: Biconditional Intro and Elimination ($<$-$>$)

i	$\vert\ \vert\ \mathcal{A}$	:AS for $\equiv$I
	$\vert\ \vert\ \vdots$	
j	$\vert\ \vert\ \mathcal{B}$	
	$\vert\ --$	
k	$\vert\ \vert\ \mathcal{B}$	:AS for $\equiv$I
	$\vert\ \vert\ \vdots$	
l	$\vert\ \vert\ \mathcal{A}$	
	$\vert\ \mathcal{A} \equiv \mathcal{B}$	:i-j, k-l $\equiv$I

Biconditional Elimination ($\equiv$E)

m	$\vert\ \mathcal{A} \equiv \mathcal{B}$	
n	$\vert\ \mathcal{A}$	
	$\vert\ \mathcal{B}$	:m, n $\equiv$E

m	$\vert\ \mathcal{A} \equiv \mathcal{B}$	
n	$\vert\ \mathcal{B}$	
	$\vert\ \mathcal{A}$	:m, n $\equiv$E

7. Midterm Review!

7. Midterm Review!

a. Recursive Definitions

Recursive Definitions have THREE clauses

- Consider an arbitrary recursively-defined set $\mathcal{S}$

1. **Base clause**: typically the smallest element(s) of your set, e.g. the atomic sentences
2. **Recursion clause**: If s is an element in $\mathcal{S}$, then so is s acted on by some basic operation(s) resulting in elements of *the next biggest* ***allowed*** *size*.
3. **Closure clause**: Nothing else is an element of $\mathcal{S}$

Palindromes semordnilaP

- A **palindrome** is a string of letters that reads the same way backwards and forwards e.g. “racecar”.
- some trivial cases: “I”, “a” (we’re thinkin ‘base case material!’)
- Consider the set of all strings in the alphabet $\{a, b, c\}$

Recursively Defining Palindromes

- Define the set of palindromes using a recursive definition:
- Call a string over $\{a, b, c\}$ a **recursive palindrome** if it satisfies the three conditions:
  1. **Base clause**: 'a', 'b', 'c', 'aa', 'bb', 'cc' are recursive palindromes
  2. **Recursion clause**: If s is a recursive palindrome, then $a * s * a$, $b * s * b$, and $c * s * c$ are recursive palindromes
  3. **Closure clause**: Nothing else is a recursive palindrome in $\{a, b, c\}$

What happens if we place restrictions on palindromes?

- Consider the alphabet $\{i, e, t\}$
- Informally define the "$tieeit$–palindromes" as all those palindromes with '$tieeit$' in the middle, with string–length divisible by 6, starting with 't', with 'e' as every third letter until the middle $tieeit$, and no consecutive i's or triple t's or e's, or instances of 'iee' or 'ite'.
  Recursively define these bad boys:
  1. **Base clause**: '$tieeit$' is a $tieeit$–palindrome
  2. **Recursion clause**: If s is a $tieeit$–palindrome, then $tie * s * eit$, $tee * s * eet$, and $tte * s * ett$, are $tieeit$–palindromes.
  3. **Closure clause**: Nothing else is a $tieeit$–palindrome (I hope!)

7. Midterm Review!

b. Induction on Strings

Complete Induction (a.k.a. ‘strong induction’)

- We do induction on string length, considering members from a recursively defined set (e.g. palindromes over some alphabet)
  Base Case: Prove the property holds in the base case(s) (of the recursive definition—elements of shortest string length(s))
  Induction Hypothesis: Assume that the property holds for members of length n, where base-case-index $\leq n < k$.
  Induction Step: Consider an arbitrary member Δ of length $k >$base case length(s). Show that Δ has the property of interest.
- Note that there must be a smaller string δ such that $\Delta = blah * \delta * blah'$, where the ‘blah’s arise from the recursion clause. (all the ways of making a longer Δ from a shorter δ)
- Remember to note that since δ has length $< k$, it falls within the induction hypothesis and hence has the property of interest.

Example: Prove by induction that...

- Show: any *tieeit*−palindrome has string length divisible by 6.
  Base Case: the base case has a single element, namely '*tieeit*' of string length 6, which is clearly divisible by 6.
  Induction Hypothesis: Assume that every *tieeit*−palindrome of length n, where $6 \leq n < k$, has the property, i.e. is divisible by 6. Show that the property holds for any *tieeit*−palindrome of length $k > 6$.
  Induction Step: consider an arbitrary *tieeit*−palindrome Δ of length $k > 6$. Then there must exist a shorter *tieeit*−palindrome δ such that Δ equals either $tie * \delta * eit$, $tee * \delta * eet$, or $tte * \delta * ett$. Since δ has length $< k$, the string length of δ is divisible by 6 by the induction hypothesis. In each case, we add 6 letters to δ to form Δ. Hence, the string length of Δ is also divisible by 6.

7. Midterm Review!

c. Induction on SL Sentences

Special Induction Schema for the language *SL*

- Recall that for SL, we can use a special induction schema:
- If you want to prove that **ALL** sentences (wffs) of SL have a particular property, it suffices to show the following:
  1. All **atomic** sentences have that property (‘**base case**’)
  2. If Φ and Ψ are two **arbitrary wffs** with that property, then so are the sentences built out of them by adding a connective
     ‘**Induction step**’ (There are five cases to consider:)
     (i) $\sim\Phi$
     (ii) $(\Phi \,\&\, \Psi)$
     (iii) $(\Phi \vee \Psi)$
     (iv) $(\Phi \supset \Psi)$
     (v) $(\Phi \equiv \Psi)$

Example: Induction on SL, using disjunctive syllogism

Prove the following by induction on wffs of SL. Don’t forget to explicitly state the **base case** and the **induction step**!

1. If a wff doesn’t contain any binary connectives, then it is contingent.
2. hint: say that a wff is *baller* if it either contains a binary connective or is contingent.
   - Use induction to show that every wff is baller.
3. Final step (disjunctive syllogism): if (i) every wff is baller and (ii) a given wff doesn’t contain any binary connectives, then (iii) it must be baller in virtue of being contingent.

Baller Induction: Base Case

Base case: this comprises the atomic sentences. So consider an arbitrary atomic sentence.

- Note that it is contingent: there exists a TVA where it is true, and a distinct TVA where it is false.
- Hence, every member in the base case is baller

Baller Induction: Induction Step

Induction Step (using our 'lazy SL' schema): consider two arbitrary well-formed formula Φ and Ψ that have the property, i.e. are baller. Show that any way of forming a more complex wff from these two also has the property.

i. (a) If Φ is contingent, then so is its negation (its negation is true whenever Φ is false and false whenever Φ is true).
(b) If Φ has a binary connective, then so does its negation $\sim\Phi$.

ii.-v. Then, consider the four cases coming from connecting Φ and Ψ with a binary connective (& , $\vee$, $\supset$, or $\equiv$). Clearly, each of these wffs has a binary connective and so is baller.

- Hence, every wff is baller.

► Hence, if a well-formed formula does not contain any binary connectives, since it is still baller, it must be contingent (disjunctive syllogism)

Non-lazy Induction Schema for SL

Induction Hypothesis: assume that every SL wff of string-length n, where $1 \leq n < k$ has the property, i.e. is baller. Show that an arbitrary wff of length k is baller.

Induction Step: Consider an arbitrary SL wff Δ of length k. Then by the recursive definition of SL wffs, there must exist shorter wffs $\mathit{\Phi}$ and $\mathit{\Psi}$ such that Δ equals either (i) $\sim\mathit{\Phi}$, (ii) $\mathit{\Phi} \,\&\, \mathit{\Psi}$, (iii) $\mathit{\Phi} \vee \mathit{\Psi}$, (iv) $\mathit{\Phi} \supset \mathit{\Psi}$, or (v) $\mathit{\Phi} \equiv \mathit{\Psi}$.

- Proceed to show that in each case, Δ has the property, i.e. is baller.

Key Fact about Conditionals

- Consider a conditional $P \supset Q$. If a truth value assignment satisfies the consequent Q, then it satisfies the conditional (regardless of the truth value of P)

7. Midterm Review!

d. Trees

Tree-contradictions and Tree-tautologies

Tree-contradiction

- A sentence Φ is a tree-contradiction if there is a tree that starts with Φ that has only closed branches
- (definition only requires the existence of a single such tree)
- (it says nothing about *all* trees starting with Φ)

Tree-tautology

- A sentence Ψ is a tree-tautology if there is a tree that starts with $\sim\Psi$ that has only closed branches
- i.e., provided that $\sim\Psi$ is a tree-contradiction
- In this case, we write '$\vdash_{STD} \Psi$'
- (again: this definition requires the existence of single such tree)

Tree-valid vs. Tree-invalid

- Consider an argument with premises given by a set Γ of wffs and conclusion Φ (i.e. Γ could be multiple sentences)
- Construct a tree with the following root: all sentences in Γ along with $\sim\Phi$ (i.e. the NEGATION of the conclusion)
- Next, apply the tree-rules until either

1.) **Each branch closes**, in which case the argument is **tree-valid**

- In this case, we write $\Gamma \vdash_{STD} \Phi$

2.) You have a complete open branch, in which case the argument is **tree-invalid**

Using trees to check for Validity

Since most homework problems follow this pattern, let's make it really explicit!

1. Add each premise to the root (number each line)
2. Add the **NEGATION** of the conclusion to the root
3. Resolve sentences until either:
    - **Each branch closes**, in which case the argument is **valid**
    - You have a complete open branch $\Rightarrow$ the argument is **invalid**

Don't forget to **justify each new node** by citing the line you are resolving and the rule you are applying

Remember that a branch closes whenever a sentence and its negation appear in its nodes (these need not be atomic sentences)

Using trees to check Tautologies

Likewise for whether a sentence is a tautology:

1. Add the **NEGATION** of the sentence to the root
2. Resolve sentences until either:
   - **Each branch closes**, in which case the sentence is **tautologous** (semantic aside: it's impossible to make the sentence false)
   - You have a complete open branch, in which case the sentence is NOT a tautology (semantic aside: it is possible to satisfy the sentence's negation, so it's possible to make the sentence in question false)

Trees: easy things to forget

- Remember to number each line, i.e. each horizontal row of sentences gets its own line number
- Remember to justify each child node by citing the line number of the sentence you are resolving and listing the rule you are using
- Remember to justify any closed branches ($\times$) by citing the line numbers of a sentence and its negation on that branch
- Don't apply any semantic equivalencies! e.g. to resolve $\sim(\sim D \vee E)$ you write $\sim\sim D$ stacked on $\sim E$. You cannot immediately write 'D'. To get D, you would have to apply the double negation rule to $\sim\sim D$

7. Midterm Review!

e. Trees Metalogic: Testing Alternative Rules

Soundness vs. completeness

- Both soundness and completeness are if–then statements (i.e. of the form $P \supset Q$):
  - **Sound**: If $\Gamma \vdash_{STD} \Theta$, then $\Gamma \vDash \Theta$
  - **Complete**: If $\Gamma \vDash \Theta$, then $\Gamma \vdash_{STD} \Theta$
- Hence, they are logically equivalent to their contrapositives:
  - Contrapositive of $(P \supset Q)$ is $({\sim}Q \supset {\sim}P)$
  - **Soundness**: If $\Gamma \nvDash \Theta$, then $\Gamma \nvdash_{STD} \Theta$
  - **Completeness**: If $\Gamma \nvdash_{STD} \Theta$, then $\Gamma \nvDash \Theta$

Some Definitions

- "$\Gamma \nvdash_{STD} \Theta$" means that *it is not the case that* the argument from Γ to Θ is **tree-valid**
- i.e., *it is not the case that* there exists a tree with root $\Gamma \cup \{\sim\Theta\}$ that possesses **all closed branches**
- Equivalently: ANY tree with root $\Gamma \cup \{\sim\Theta\}$ possesses
  at least one complete open branch
- (Aside: this is NOT the same as saying that the argument is **tree-invalid**, since that only requires the existence of a single tree with a complete open branch)

Modifying system STD

- What if we had chosen an alternative rule(s)?
- Simplest case: we swap out one of our nine rules for a different rule, i.e. for a given connective or negated connective.
- Call our modified system ‘STD^*’
- Would our modified system STD^* remain Sound?
- Would our modified system STD^* remain Complete?

Some Study Advice

- I recommend writing out on a ‘cheat sheet’ the following steps for checking whether a modified rule preserves (i) soundness or (ii) completeness
- In each case, note on your sheet the method for extending the (i) soundness or (ii) completeness proof
- Note the method for constructing a counterexample, i.e. what it would take to have (i) a counterexample to soundness or (ii) a counterexample to completeness
- You probably want to have this information right in front of you during the exam, rather than to be scrambling looking for it

Checking Soundness: “top-down” reasoning

- **Heuristic for Soundness checking**: consider an **arbitrary** truth value assignment (TVA) that satisfies the sentence being resolved (i.e. *makes true* the sentence ‘at the top’ of the rule)
- Ask whether this TVA guarantees that **at least one** child-node is satisfied, i.e. the sentences on that child-node are all made true
- If ‘**yes**’, then the rule preserves soundness: proceed to extend our soundness proof for this case (reasoning in terms of a TVA ‘$\mathcal{I}$’).
- If ‘**no**’, then the rule BREAKS soundness:
  proceed to **construct a counter-example** using the rule

Counterexamples to Soundness

- To show soundness fails, it is NECESSARY to provide a CONCRETE counterexample, using SL sentences
  - Further heuristic reasoning about TVAs is not enough!

- What you need for a counterexample to soundness:
  1.) Choose a satisfiable root, i.e. a **consistent** set of sentences

  2.) Apply the modified rule in the tree; you may also use other rules that we've already shown preserve soundness

  3.) Show that the tree **closes** (i.e. NO complete open branches)

Why such Counterexamples work

- **Sound**: If $\Gamma \vdash_{STD^*} \Theta$, then $\Gamma \vDash \Theta$
- Counterexample: $\Gamma \vdash_{STD^*} \Theta$ but $\Gamma \nvDash \Theta$:
  - '$\Gamma \vdash_{STD^*} \Theta$' means tree with root $\Gamma \cup \{\sim\Theta\}$ **CLOSES** (i.e. is tree–valid in system STD^*)
  - '$\Gamma \nvDash \Theta$' means that $\Gamma \cup \{\sim\Theta\}$ is **CONSISTENT** (i.e. satisfiable) (i.e. there exists a TVA that makes the premises Γ true but the conclusion Θ false)
- If our system *were* sound, then whenever a tree closes, it *would* correspond to a semantically valid argument.

Checking Completeness: “bottom-up” reasoning

- **Heuristic for Completeness checking**: ask whether **EACH** child node (below) individually entails the sentence being resolved (i.e. the sentence ‘up top’, in the ‘parent-node’)
- If ‘**yes**’, then the rule preserves completeness: proceed to extend our completeness proof for this case (reasoning in terms of arbitrary TVA’s, one for each child node).
- If ‘**no**’, then the rule BREAKS completeness: proceed to **construct a counter-example** using the rule.

Counterexamples to Completeness

- To show completeness fails, it is NECESSARY to provide a CONCRETE counterexample, using SL sentences
  - Further heuristic reasoning about TVAs is not enough!

- What you need for a counterexample to completeness:
  1.) Choose an **UNsatisfiable root**, i.e. an INconsistent set of sentences
  2.) Apply the modified rule in the tree; you may also use other rules that we've already shown preserve completeness
  3.) Show that the tree has a **complete open branch**, i.e. does NOT close

Why such Counterexamples work

- **Complete**: If $\Gamma \vDash \Theta$, then $\Gamma \vdash_{STD^*} \Theta$
- Counterexample: $\Gamma \vDash \Theta$ but $\Gamma \nvdash_{STD^*} \Theta$
  - '$\Gamma \vDash \Theta$' means that $\Gamma \cup \{\sim\Theta\}$ is INconsistent (i.e. UNsatisfiable) (i.e. any TVA that makes Γ true makes the conclusion Θ true)
  - '$\Gamma \nvdash_{STD^*} \Theta$' means that it is NOT the case that the argument is tree–valid in STD^* (a claim about ALL trees)
  - From completeness proof, we know that if the system *were* complete, then we could use any **complete open branch** to construct a TVA that **satisfies** the root
  - Hence, if we construct a **complete open tree** with an **unsatisfiable root**, this is a reductio of completeness

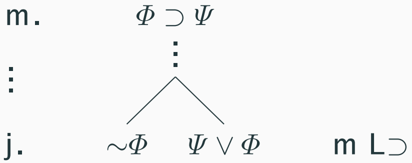
A Common Mistake to Avoid!!!!

- You are very liable to forget to COMPLETE YOUR open branch!!!
- You only have a counterexample to completeness if you have a COMPLETE open branch with an unsatisfiable root
- So make sure you FULLY RESOLVE every sentence on the open branch, until you can write that coveted up arrow ‘$\uparrow$’!
- When in doubt, just complete the whole tree

Liberal Conditional and Conservative Biconditional

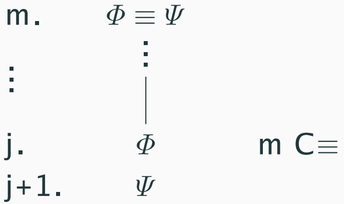
STD*: replace rule ($\supset$) with:

Liberal Conditional (L$\supset$)



STD†: replace rule ($\equiv$) with:

Conservative Biconditional (C$\equiv$)



Conservative Biconditional

- Reason from the top-down:
  $\Phi \equiv \Psi$ is logically equivalent to $(\Phi \,\&\, \Psi) \vee ({\sim}\Phi \,\&\, {\sim}\Psi)$

- The node below is logically equivalent to $(\Phi \,\&\, \Psi)$

- Note that the top does NOT entail the bottom! Given an interpretation that satisfies $\Phi \equiv \Psi$, it is NOT guaranteed to satisfy the sentence(s) in at least one branch below.

- Hence, **to the counterexample!**
  (note that the problem REQUIRES this! can't stop won't stop!)

Counterexample to Soundness of $STD^{\dagger}$

- For a counterexample, need: $\Gamma \vdash_{STD^{\dagger}} \Theta$ but $\Gamma \nvDash \Theta$

  $STD^{\dagger}$: replace rule ($\equiv$) with *Conservative Biconditional* (C$\equiv$):

1.	$P \equiv P$	Premise (PR)
2.	$\sim P$	$\sim$Conclusion
	$\vert$	
3.	P	1 C$\equiv$
4.	P	
	$\times$	
	2, 3, so tree-valid in $STD^{\dagger}$	

- But $(P \equiv P) \nvDash P$ because $\{(P \equiv P), \sim P\}$ is consistent! Assign ‘P’ false! N.B.: your counterexample MUST use actual sentences of SL; not meta-variables!

Conservative Biconditional (bottoms up!)

- Reason from the bottom up: there is only one child node. So we only need its sentences to entail $\Phi \equiv \Psi$
- Sentences on bottom are equivalent to $(\Phi \& \Psi)$, which DOES entail $\Phi \equiv \Psi$. So we proceed to formally extend our completeness proof:
  - **Formally**: since only one child node, both Φ and Ψ lie on the complete open branch O.
  - By induction hypothesis, both are true according to $\mathcal{I}$.
  - Hence, $\Phi \equiv \Psi$ is true on $\mathcal{I}$, which is what we needed to show.

7. Midterm Review!

f. Translation/Symbolization in SL

Unless

- Translate ‘unless’ as ‘or’ (unless you want to make things needlessly complicated for yourself!)
- Example: I am not going to not Vote in the midterms, unless a meteor Destroys my polling place and the city of Cambridge does not give me an Alternative location.
- $\sim\sim V \vee (D \,\&\, \sim A)$

Provided that; lonely if; given: these flip the order!

- ‘Q provided that P’ is ‘If P, then Q’
- So just like the “lonely if”!
- ‘Q if P’ is ‘If P, then Q’
- Likewise for ‘Q given P’: ‘If P, then Q’

Only if

- ‘P only if Q’ is ‘If P, then Q’
- so order is preserved:
- Intuition: ‘P only if Q’ means ‘If not Q, then not P’, which is ‘$\sim Q \supset \sim P$’, i.e. a contrapositive
- A contrapositive is logically equivalent to its conditional: $P \supset Q$

Just in Case

- Translate “just in case” as “if and only if”
- Example: I will do Well in this class just in case I do not Party every night.
- $W \equiv \sim P$

‘Although’ and ‘But’

- both ‘although’ and ‘but’ are translated as ‘and’
- i.e. schematized as $\&$
- Example: Although I Meditate every day, I remain an Anxious person, but I Hope to overcome this in the future.
- $(M \,\&\, A) \,\&\, H$

7. Midterm Review!

g. Truth Tables: (In)Validity and (In)Equivalence

Proving an argument is Valid

- To use a truth table to prove that an argument is valid, one must:
- Complete the entire truth table!
- One must show that there is no truth value assignment that makes all of the premises true while making the conclusion false (i.e. no counterexamples to validity)
- Equivalently: every TVA that makes the premises true makes the conclusion true
- (note that this ‘every’ claim is vacuously true if your conclusion is a tautology, since a tautology is not false on any TVA)

Proving an argument is Invalid

- To prove that an argument is invalid, it suffices to:
- Identify at least one truth value assignment that makes the premises true but the conclusion false
- In this case, unnecessary to complete the entire truth table
- *Carnap* will let you enter the relevant TVA as a counterexample to validity (make sure you understand the syntax for this!)

Proving that two sentences are Equivalent

- Using a truth table, to prove that two sentences are equivalent:
- You must complete the entire table!
- You must show that the two sentences receive the same truth value for every possible truth value assignment (i.e. every row of the truth table)

Proving that two Sentences are Inequivalent

- Suffices to identify a single TVA where the sentences have different truth values
- In this case, unnecessary to complete the entire truth table
- *Carnap* will let you enter the relevant TVA as a counterexample to equivalence

7. Midterm Review!

h. Natural Deduction

Some rules to Definitely Understand

- Disjunction Elimination
- Conditional Introduction
- Negation introduction; Negation Elimination
- Understand when and how to start a subproof
- For rules that cite a subproof, remember to use a HYPHEN between line numbers in the justification (you are citing the ENTIRE subproof, even in those cases where the sub proof is itself only 2 lines long)

Using Conditional Introduction $\supset I$

- If you need to construct a conditional, typically you assume the antecedent, starting a subproof (tab in and write :AS after the antecedent)
- Derive the consequent within the subproof
- Exit the subproof by writing the conditional and justifying with $\supset I$

Common Mistake with Negation Rules

- Don't forget that you need both a sentence Ψ and $\sim\Psi$ WITHIN the subproof, i.e. under the assumption line
- Often, if you are using a sentence that has already occurred as one of the Ψ or $\sim\Psi$, you will need to reiterate it on a line within the subproof
- *Carnap* seems to be a bit fussy: the Ψ and $\sim\Psi$ cannot themselves be separated by a subproof within your subproof

8. Intro. to Quantifier Logic

8. Intro. to Quantifier Logic

a. The goals of QL

Limits of symbolization in SL

- Consider the argument:

  Greta is a hero.

  ∴ There is a hero.
- It's clearly valid: in any case in which Greta is a hero, someone (or something, at least) is a hero, so there must be a hero.
- But its symbolization in SL is invalid in SL:

  G

  $\therefore H$

The problem

- Symbolization in SL allows us to break down sentences containing “and,” “or,” “if-then” and determine validity in virtue of these **connectives**.
- Anything that can’t be further broken down must be symbolized by sentence letters.
- That includes basic sentences like “Greta is a hero,” but also:
  - Everyone is a hero.
  - No one is a hero.
  - All heroes wear capes.

The goals of Quantifier Logic (QL)

- Finer-grained symbolization
- Augments SL (all of SL and *more*!)
- Allows for precise semantics (like truth tables for SL)
- Works with natural deduction (add new rules!)
- Be simple & expressive (only a few new symbols!)

The goals of QL

- Consider the valid argument:

  Greta is a hero.

  Greta does not wear a cape.

  ∴ Not all heroes wear capes.
- We'll need to connect the occurrences of the name "Greta" in the premises
- We'll need to connect "hero" in the premise and conclusion
- We want to retain using the symbol '$\sim$' for "not"
- Ultimately, we'll want our argument–symbolization to have a proof

8. Intro. to Quantifier Logic

b. Beginning symbolization in QL

First steps: names (a.k.a. ‘constants’)

- Purpose of a **proper name**: to pick out a single, specific thing.
- (Contrast with common nouns like “hero” or “rock” which pick out collections of things)
- For simplicity, we’ll only consider names that pick out a **specific object** (often within a hypothetical case we’re considering)
- Later on, we’ll be able to deal with other expressions that play a similar role to names, e.g., “the president of the USA”
- In QL, names are symbolized by lowercase letters a-v (allowing natural number sub-scripts, e.g. m_1, t_{2022})

First steps: predicates (including properties and relations)

- Remove a name from a sentence. What's left over is a **predicate**:

  Greta is a hero

  $\Rightarrow$ $\_\_\_\_\__x$ is a hero. (i.e. property of being a hero)

  Greta admires Autumn

  $\Rightarrow$ $\_\_\_\_\__x$ admires $\_\_\_\_\__y$. (i.e. relation of x admiring y)
- In QL, predicates are symbolized using uppercase letters *A*-*Z* plus a number of argument slots (marked with variables), e.g., *Hx* or *Axy*.
- Argument slots correspond to blanks.

Symbolization keys

- **Names/constants**: lowercase letters for proper names
- **Predicates**: uppercase letters with variables marking blanks

a:	Autumn
g:	Greta
Hx:	$\_\_\_\_\__{x}$ is a hero
Vx:	$\_\_\_\_\__{x}$ is a villain
Ix:	$\_\_\_\_\__{x}$ inspires
Cx:	$\_\_\_\_\__{x}$ wears a cape
Wxy:	$\_\_\_\_\__{x}$ welcomed $\_\_\_\_\__{y}$
Axy:	$\_\_\_\_\__{x}$ admires $\_\_\_\_\__{y}$
Yxy:	$\_\_\_\_\__{x}$ is younger than $\_\_\_\_\__{y}$

- **Domain**: the non-predicate objects we're talking about in a context—also called grandly the '**Universe of Discourse**' (UD)

  e.g., people alive in 2022

Symbolization of Sentences without Quantifiers

- Basic sentences: predicates with names replacing variables.
  - Greta is a hero: *Hg*
  - Greta admires Autum: *Aga*
- Combinations using connectives:
  - Greta and Autumn are heroes: $Hg \mathbin{\&} Ha$
  - If Autumn admires Greta, then Autumn is a hero: $Aag \supset Ha$

Symbolization of Pronouns

- Replacing pronouns by antecedents:
  - If Autumn is a hero, Greta admires **her**: $Ha \supset Aga$
  - Greta doesn't admire **herself**: $\sim Agg$ (but she should!)
  - Greta and Autumn welcomed **each other**: $Wga \,\&\, Wag$

- Modifiers:
  - Autumn is an **inspiring hero**:
    i.e. Autumn inspires, and she is a hero: $Ia \,\&\, Ha$
  - Greta is a **hero who doesn't wear a cape**:
    Greta is a hero, and it's not the case that Greta wears a cape:
    $Hg \,\&\, \sim Cg$

Mind the Modifiers!

- ‘Greta is an international hero’:
  - Can’t be paraphrased as
    “Greta is international and a hero.”

  - So “_____ is an international hero” needs its own predicate
- ‘The Piltdown Man is a fake fossil’
  - Can’t be paraphrased as
    “The Piltdown Man is fake and a fossil.”

  - Since “fake” and other privative adjectives (“pretend,” “fictitious”) deny the property that they modify! (‘fake news’ isn’t news!)

Examples

- Autumn and Greta are inspiring heroes.
  $(Ia \& Ha) \& (Ig \& Hg)$
- Greta admires Autumn but not herself.
  $Aga \& {\sim}Agg$
- Greta inspires only if Autumn does.
  $Ig \supset Ia$
- Greta and Autumn welcomed each other.
  $Wga \& Wag$
- Greta is older than Autumn.
  Yag (i.e. Autumn is *younger than* Greta)
- One of Greta and Autumn welcomed the other.
  At least one: Exactly one:
  $Wga \lor Wag$ $(Wga \lor Wag) \& {\sim}(Wga \& Wag)$

8. Intro. to Quantifier Logic

c. The existential quantifier

Existential quantifier (something or other)

- In English: “something,” “someone,” “there is . . . ”
- For instance:
    - **Someone** wears a cape.
    - **There is** a hero.
    - **Something** inspires.

- Note: often goes where names and pronouns are placed
- But works differently from names (“something” doesn’t pick out a unique, specific object).

How (not) to symbolize "something"

- Idea(?): introduce a special term '*sg*' for 'a something'?
- Problem: now we can't distinguish between
  - Someone is a hero and wears a cape.
  - Someone is a hero and someone wears a cape.

  as both would be symbolized by '$H(sg) \& C(sg)$'.
- Better idea: symbolize (complex) **properties** and introduce a notation for expressing that properties are **instantiated**

Expressing properties (and relations)

- One-place predicates **express** properties, e.g.,
  - *Hx* expresses property "being a hero"
  - *Ix* expresses "is inspiring" ('x' is a variable)
- Combinations of predicates (with connectives, names) can express **derived** properties, e.g.,
  - *Axg* expresses "admires Greta"
  - *Wax* expresses "is welcomed by Autumn"
  - *Hx* & *Cx* expresses "is a hero who wears a cape"
- Note: all contain a **single** variable *x*

The existential quantifier $\exists$

- Symbol for "there is": $\exists$
- Combine '$\exists$' with an expression for a property (e.g., $(Hx \& Cx)$) to say "something (or someone) has that property"
- Put the variable that serves as a marker for the gap after $\exists$. E.g.,

  $$(\exists x)(Hx \& Cx)$$

  says "Someone is a hero and wears a cape"
- MUST always wrap a quantifier and the variable it 'binds' within **parentheses**: $(\exists y)$; *Carnap* will require this!!!

Quantifiers and variables

$$\text{Compare: } (\exists x)\,(Hx \,\&\, Cx) \text{ to}$$
$$(\exists x)\,Hx \,\&\, (\exists x)\,Cx$$

- In first case, *the same person* must be a hero and wear a cape.
- In second case, one person can be the hero and another (possibly different) person wears a cape.
- Instances of ‘$(\exists x)$’ separated by other connectives are independent, even if they bind the same variable x.
  e.g. there’s no difference in meaning between the following:

$$(\exists x)\,Hx \,\&\, (\exists x)\,Cx \quad \textbf{vs.}$$
$$(\exists x)\,Hx \,\&\, (\exists y)\,Cy$$

- But we’ll never write ‘$(\exists x)(\exists x)(Hx \,\&\, Cx)$’

The domain (UD) and quantifiers

- Symbolization key gives a domain of objects being talked about.
- Quantifier **ranges over** this ‘universe of discourse’ (UD).
- That means: $(\exists x) \ldots x \ldots$ is true iff some object **in the domain** has the property expressed by $\ldots x \ldots$.
- Domain makes a difference: Consider $(\exists x)\, Wxg$.
  - True if someone welcomed Greta (say, Autumn did).
  - Now take the domain to include only Greta.
  - Relative to that domain, $(\exists x)\, Wxg$ is true iff Greta welcomed herself (e.g. to the left-over chocolate fondue!)

Quantifier restriction in English

- "something" and "someone" work grammatically like singular terms (they can go where names can also go).
- "some" (on its own) does not: it is a **determiner** and needs a **complement**, e.g.,
  - a common noun ("some hero"), or
  - a noun phrase ("some admirer of Greta").
- "some" + complement works grammatically like "someone", e.g., "**Some hero** wears a cape"
- General form: "Some *F* is *G*."

Quantifier restriction in QL

- "Some F is G" **restricts** the "something" quantifier to Fs.
- We could (and linguists often do) mark restrictions in the quantifier, e.g., $((\exists x) : Fx)Gx$
- We won't because we can do without this additional notation
- "Some F is G" is true iff there is something which is both F and also G, so:
- "Some F is G" can be symbolized as

$$(\exists x)(Fx \,\&\, Gx)$$

- We'll also symbolize the plural form this way ("Some Fs are Gs").
- And more generally (most) sentences of the form: "G(some F)" or "G(something that Fs)".

Examples

- Some hero wears a cape.
  Some heroes wear capes.

  $(\exists x)(Hx \mathbin{\&} Cx)$
- Someone who wears a cape welcomed Greta.

  $(\exists x)(Cx \mathbin{\&} Wxg)$
- Greta admires some hero who wears a cape.

  $(\exists x)((Hx \mathbin{\&} Cx) \mathbin{\&} Agx)$
- Autumn welcomed someone who welcomed Greta.

  $(\exists x)(Wxg \mathbin{\&} Wax)$

8. Intro. to Quantifier Logic

d. The universal quantifier

Universal quantifier

- “**Something** is F” is true iff **at least one** element of domain is F.
- “**Everything** is F” is true iff **every element** of the domain is F.
- In QL: $(\forall x)\ Fx$.
- E.g.:
  - “Everyone wears a cape”: $(\forall x)\ Cx$
  - “Everyone welcomed Greta or Autumn”: $(\forall x)(Wxg \vee Wxa)$

Universal determiners: all, every, any

- Determiners with universal meaning: **all, every, any**.
- Take complements (just like “some” does), e.g.,
  - Every hero inspires.
  - All heroes inspire.
  - Any hero inspires.
- These are true in the same cases (i.e. they are synonymous).
- “Every *F* is *G*” is true iff everything which is an *F* is *G*.
- Watch out for “any”: not always universal.

Restricted $\forall$ in QL

- Suppose we can symbolize two properties 'F' and 'G'.
- How do we symbolize "Every F is G"?
- Initial Ideas (only one of which is correct):
  - $(\forall x)(Fx \mathbin{\&} Gx)$
    If true, everything must be F.
    So can be false when "Every F is G" is true.
  - $(\forall x)(Fx \vee Gx)$
    True if everything is F (without being G).
    So can be true when "Every F is G" is false.
  - $(\forall x)(Fx \supset Gx)$
    If x is F, x must also be G.
    (If x is not F, doesn't matter if it's G or not.)

Symbolizing “all Fs are Gs” (memorize this!)

Symbolize the following as

$$(\forall x)(Fx \supset Gx)$$

- All Fs are Gs.
- Every F is G.
- Any F is G.

Examples

- **Every hero** wears a cape.
  All heroes wear capes.
  $(\forall x)(Hx \supset Cx)$
- **Every hero who wears a cape** welcomed Greta.
  $(\forall x)((Hx \,\&\, Cx) \supset Wxg)$
- Greta and Autumn admire **anyone who wears a cape**.
  $(\forall x)(Cx \supset (Agx \,\&\, Aax))$
- Autumn welcomed **everyone who welcomed Greta**.
  $(\forall x)(Wxg \supset Wax)$
- All heroes and villains welcomed Greta (a tricky one!).
  $(\forall x)((Hx \vee Vx) \supset Wxg)$
  equivalent to $(\forall x)((Hx \supset Wxg) \,\&\, (Vx \supset Wxg))$

8. Intro. to Quantifier Logic

e. ‘No’, ‘only’, ‘a’, ‘some’, and ‘any’

No *F* is *G*

- "**No *F*s are *G*s**" can be paraphrased as
  - "**Every** *F* is **not**–*G*," or as
  - "**Not: some** *F* is *G*."
- So symbolize it using:
  - $(\forall x)(Fx \supset \sim Gx)$ or
  - $\sim(\exists x)(Fx \,\&\, Gx)$ (i.e. 'it is not the case that there is something that is both an F and a G')

Examples

- **No hero** wears a cape.
  No heroes wear capes.

  $(\forall x)(Hx \supset {\sim}Cx)$

- **No hero who wears a cape** welcomed Greta.

  $(\forall x)((Hx \mathbin{\&} Cx) \supset {\sim}Wxg)$

- Greta admires **no one who wears a cape**.

  ${\sim}(\exists x)(Cx \mathbin{\&} Agx)$

- Autumn welcomed **no one who welcomed Greta**.

  ${\sim}(\exists x)(Wxg \mathbin{\&} Wax)$

Only *F*s are *G*

- When is “Only *F*s are *G*s” false?
- When there is a **non**−*F* that is a *G*.
- So symbolize it as

$$\sim(\exists x)(\sim Fx \mathbin{\&} Gx)$$

- Or, paraphrase it as: “Any *x* is *G* **only if** it is *F*”
- So another symbolization is:

$$(\forall x)(Gx \supset Fx)$$

i.e. being an F is a necessary condition for being a G:
if it’s not an F, then it can’t be a G

Examples

- Only heroes wear capes:

  $(\forall x)(Cx \supset Hx)$ (being a hero is necessary for cape-wearing)

- Only heroes who wear capes welcomed Greta.

  $(\forall x)(Wxg \supset (Hx \,\&\, Cx))$

- Greta admires only people who wear capes.

  $\sim(\exists x)(\sim Cx \,\&\, Agx)$

- Autumn welcomed only heroes and villains.

  $\sim(\exists x)(\sim(Hx \vee Vx) \,\&\, Wax)$

  $(\forall x)(Wax \supset (Hx \vee Vx))$

The indefinite article

- We use "is a" to indicate predication, e.g., "Greta is a hero."
- Often "a" is used to claim existence, e.g.,

  Greta admires a hero.

  $(\exists x)(Hx \,\&\, Agx)$
- But a **generic** indefinite is closer to a universal quantifier:

  A hero is someone who inspires.

  $(\forall x)(Hx \supset Ix)$
- Be careful if the indefinite article is in the antecedent of a conditional:

  If **a hero** wears a cape, **they** inspire.

  That means: all heroes who wear capes inspire.

  $(\forall x)((Hx \,\&\, Cx) \supset Ix)$

Universal “some”; existential “any”

- “Someone,” “something” can require a **universal** quantifier: if it’s in the antecedent of a conditional, with a pronoun in the consequent referring back to it, e.g.,

  If **someone** is a hero, Autum admires **them**.

  Roughly: Autumn admires all heroes.

  $(\forall x)(Hx \supset Aax)$

- “Any” in antecedents but **without** pronouns referring back to them are **existential**:

  If **anyone** is a hero, Greta is.

  Roughly: if there are heroes (at all), Greta is a hero.

  $(\exists x)\, Hx \supset Hg$

8. Intro. to Quantifier Logic

f. Mixed domains

Mixed domains

- Sometimes you want to talk about more than one kind of thing.
- The domain can include any mix of things (e.g., people, animals, items of clothing, feelings)
- Proper symbolization then needs predicates for these kinds, e.g.:

Domain: people alive in 2022 and items of clothing

Px: $\_\_\_\_\__x$ is a person

Lx: $\_\_\_\_\__x$ is an item of clothing.

Ex: $\_\_\_\_\__x$ is a cape (recall: 'Cx' is 'wears a cape')

Rxy: $\_\_\_\_\__x$ wears $\_\_\_\_\__y$

Quantification in mixed domains

- Not everyone is wearing a cape.
  - In domain of people only:
    $\sim(\forall x)\ Cx$
  - In mixed domain:
    $\sim(\forall x)\ (Px \supset Cx)$
- Some people inspire.
  - In domain of people only:
    $(\exists x)\ Ix$
  - In mixed domain:
    $(\exists x)(Px \,\&\, Ix)$
- Greta wears something.
  $(\exists x)(Lx \,\&\, Rgx)$

8. Intro. to Quantifier Logic

g. Captain Morgan’s (tele)scope!

(Spiced) de Morgan's for Quantifiers

- We can push negations through quantified expressions, flipping the quantifiers and negating what lies in their scope:
- $\sim(\forall\chi)\Phi(\chi)$: "it's not the case that for everything, Phi"
  is equivalent to
  $(\exists\chi)\sim\Phi(\chi)$: "there exists something such that not–Phi"
- $\sim(\exists\chi)\Phi(\chi)$: "it's not the case that for something, Phi"
  is equivalent to
  $(\forall\chi)\sim\Phi(\chi)$: "for everything, not–Phi", i.e. Phi for–nothing!
  A concrete example:
- *It's not the case that some hero wears a cape*:
  $\sim(\exists x)(Hx \,\&\, Cx)$
  $(\forall x)\sim(Hx \,\&\, Cx)$ (now apply regular de Morgan's!)
  $(\forall x)(\sim Hx \vee \sim Cx)$

Quantifier Scope

- **Scope of a Quantifer**: the **sub**formula for which the quantifier is the main logical operator

  in '$(\exists x)(Lx \& Rgx)$', the scope of the existential is '$(Lx \& Rgx)$'

  in '$(\exists x)Lx \& (\forall y)Rgy$', the scope of the existential is 'Lx'

- A variable is **bound** if it lies within the scope of a quantifer
- By the recursive definition that follows, each variable is bound by at most one quantifier!

Recall our Recursive definition of SL wffs

1. Every atomic formula is a wff.
2. If $\mathit{\Phi}$ is a wff, then $\sim\mathit{\Phi}$ is a wff.
3. If $\mathit{\Phi}$ and $\mathit{\Psi}$ are wffs, then $(\mathit{\Phi} \,\&\, \mathit{\Psi})$ is a wff.
4. If $\mathit{\Phi}$ and $\mathit{\Psi}$ are wffs, $(\mathit{\Phi} \vee \mathit{\Psi})$ is a wff.
5. If $\mathit{\Phi}$ and $\mathit{\Psi}$ are wffs, then $(\mathit{\Phi} \supset \mathit{\Psi})$ is a wff.
6. If $\mathit{\Phi}$ and $\mathit{\Psi}$ are wffs, then $(\mathit{\Phi} \equiv \mathit{\Psi})$ is a wff.

Nothing else is a wff of SL! (But some new things are wffs of QL!)

Extending our Recursive definition to QL wffs

1.* **Atomic formula of QL**: an n–place predicate followed by n terms (i.e. constants or variables), where $n \in \mathbb{N}$

   This includes the SL atomic wff, which are 0–place predicates

7. If Φ is a wff, χ is a variable, and Φ contains no χ–quantifiers, then $(\forall \chi)\Phi$ is a wff.

8. If Φ is a wff, χ is a variable, and Φ contains no χ–quantifiers, then $(\exists \chi)\Phi$ is a wff.

9. All and only wffs of QL come from the prior 8 rules.

QL Sentences: proper subset of QL wffs

- Recall: the wffs of SL *just are* the sentences of SL, i.e. the statements that are true or false under a truth-value assignment to atomic sentences
- Not all wffs of QL are sentences! Watch out!
- Let L be a two-place predicate. Then Lxx is an atomic wff of QL, but NOT a sentence (‘Lxx’ is neither true nor false)
- the x’s in ‘Lxx’ are **free variables**, i.e. unbound variables
- **Sentence of QL**: a wff that has no free variables: i.e. any variable that occurs is bound by a quantifier

9. Semantics of QL

9. Semantics of QL

a. Arguments and validity in QL

Validity of arguments

Valid?

Everyone is either good or evil.

Not everyone is a villain.

Only villains are evil.

∴ Some heroes are good.

Validity in QL

- Want to capture validity **in virtue of the meanings of the connectives and the quantifiers**
  (but ignoring the meanings of predicate symbols)
- So we want to ignore any restrictions the predicate symbols place on their **extensions**
- Hence: allow **any** extension in a potential counterexample
- An argument is **QL-valid** if there is **no interpretation** in which the premises are true and the conclusion false

Forms of arguments

Everyone is either good or evil.

Not everyone is a villain.

Only villains are evil.

$\therefore$ Some heroes are good.

$(\forall x)(Gx \vee Ex)$

$\sim(\forall x)\, Vx$

$(\forall x)(Ex \supset Vx)$

$\therefore$ $(\exists x)(Hx \,\&\, Gx)$

(In)validity of arguments

$(\forall x)(Gx \vee Ex)$

$\sim(\forall x)\ Vx$

$(\forall x)(Ex \supset Vx)$

$\therefore\ (\exists x)(Hx \,\&\, Gx)$

Domain:	the inner planets (Mecury, Venus, Mars, Earth)
Gx:	x is smaller than Earth
Ex:	x is inhabited
Vx:	x has a moon
Hx:	x has rings

9. Semantics of QL

b. Interpretations

Interpretations

- Domain: collection of objects (not empty)
- **Referents** for each name (which object it names)
- Properties of each object
  - **Extension** of each 1–place predicate symbol:
    the set of objects it applies to
- Relations between each pair of objects
  - **Extension** of each 2–place predicate symbol:
    all pairs of objects standing in that relation

Extensions

Domain:	the inner planets
Gx:	x is smaller than Earth
Ex:	x is inhabited
Vx:	x has a moon
Hx:	x has rings

Domain:	Mercury, Venus, Earth, Mars
Gx:	Mercury, Venus, Mars
Ex:	Earth
Vx:	Earth, Mars
Hx:	—

(In)validity of arguments

$$(\forall x)(Gx \vee Ex)$$

$$\sim(\forall x)\, Vx$$

$$(\forall x)(Ex \supset Vx)$$

$$\therefore\ (\exists x)(Hx \,\&\, Gx)$$

Domain:	Mercury, Venus, Earth, Mars
Gx:	Mercury, Venus, Mars
Ex:	Earth
Vx:	Earth, Mars
Hx:	—

(In)validity of arguments

$$(\forall x)(Gx \vee Ex)$$

$$\sim(\forall x)\, Vx$$

$$(\forall x)(Ex \supset Vx)$$

$$\therefore\ (\exists x)(Hx \,\&\, Gx)$$

Domain:	1, 2, 3, 4
Gx:	1, 2, 4
Ex:	3
Vx:	3, 4
Hx:	—

(In)validity of arguments

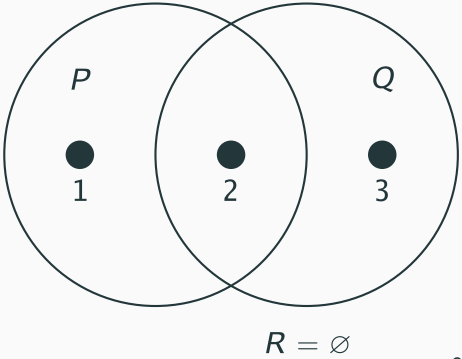
$$(\forall x)(Gx \vee Ex)$$
$$\sim(\forall x)\, Vx$$
$$(\forall x)(Ex \supset Vx)$$
$$\therefore\ (\exists x)(Hx \,\&\, Gx)$$

Domain: 1

Gx:	1
Ex:	—
Vx:	—
Hx:	—

Extensions of predicates

Domain: 1, 2, 3
Px: 1, 2
Qx: 2, 3
Rx: —



(In)validity of arguments

$(\forall x)(Gx \vee Ex)$

$\sim(\forall x)\, Vx$

$(\forall x)(Ex \supset Vx)$

$\therefore\ (\exists x)(Hx \,\&\, Gx)$

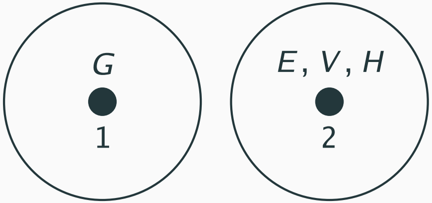
Domain: 1, 2

Gx: 1

Ex: 2

Vx: 2

Hx: 2

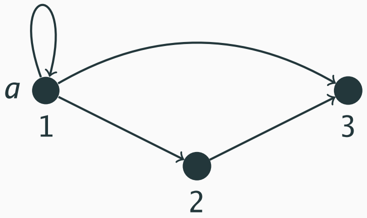


Extensions of predicates

Domain: 1, 2, 3

a: 1

Axy: $\langle 1, 1\rangle$, $\langle 1, 2\rangle$, $\langle 1, 3\rangle$, $\langle 2, 3\rangle$



9. Semantics of QL

c. Truth of sentences of QL

Truth of sentences of QL

- Given an interpretation I . . .
- An **atomic sentence** is true iff the referents of the constants are in the extension of the predicate:
  - Pa is true iff referent 'r' of a is in extension of P
  - Rab is true iff $\langle r, p\rangle$ is in extension of R
    (where r is referent of a, and p is referent of b)
- $\sim\mathcal{A}$ is true iff $\mathcal{A}$ is false
- $\mathcal{A} \vee \mathcal{B}$ is true iff at least one of $\mathcal{A}$, $\mathcal{B}$ is true
- $\mathcal{A} \,\&\, \mathcal{B}$ is true iff both $\mathcal{A}$, $\mathcal{B}$ are true
- $\mathcal{A} \supset \mathcal{B}$ is true iff $\mathcal{A}$ is false or $\mathcal{B}$ is true

Truth of quantified sentences

- $(\exists x)\,\mathcal{A}x$ is true iff $\mathcal{A}x$ is **satisfied** by **at least one** object in the domain
    - r satisfies $\mathcal{A}x$ in I iff $\mathcal{A}r$ is true in the interpretation
- $(\forall x)\,\mathcal{A}x$ is true iff $\mathcal{A}x$ is **satisfied** by **every** object in the domain

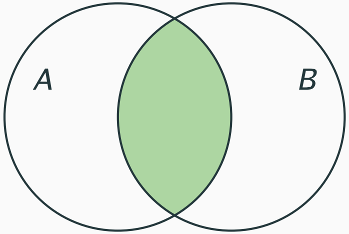
Truth of quantified sentences

- $(\exists x)(\mathcal{A}x \,\&\, \mathcal{B}x)$ is true iff some object satisfies '$\mathcal{A}x \,\&\, \mathcal{B}x$'
  - o satisfies '$\mathcal{A}x \,\&\, \mathcal{B}x$' iff it satisfies both $\mathcal{A}x$ and $\mathcal{B}x$
- $(\forall x)(\mathcal{A}x \supset \mathcal{B}x)$ is true iff **every** object satisfies '$\mathcal{A}x \supset \mathcal{B}x$'
  - o satisfies '$\mathcal{A}x \supset \mathcal{B}x$' iff either
    - o does not satisfy $\mathcal{A}x$ (vacuously true conditional)

      or
    - o does satisfy $\mathcal{B}x$

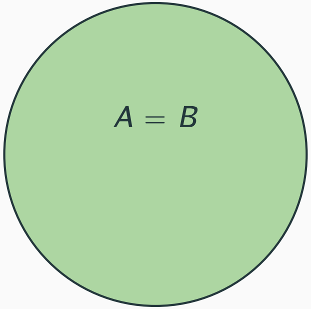
Making "Some *A*s are *B*s" true

- $(\exists x)(Ax \mathbin{\&} Bx)$
- Extension of *A* and *B* must have something in common. (Filled area must contain at least one object)
- *A* and *B* can overlap, be equal, or be contained.
- Same situations make "No *A*s are *B*s" **false**.



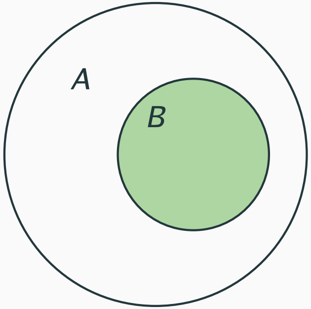
Making “Some *A*s are *B*s” true

- $(\exists x)(Ax \mathbin{\&} Bx)$
- Extension of *A* and *B* must have something in common. (Filled area must contain at least one object)
- *A* and *B* can overlap, be equal, or be contained.
- Same situations make “No *A*s are *B*s” **false**.



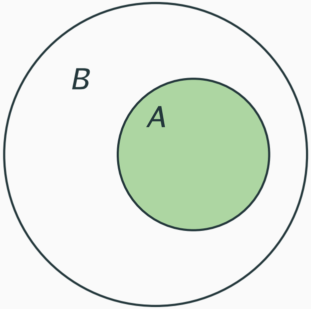
Making “Some *A*s are *B*s” true

- $(\exists x)(Ax \,\&\, Bx)$
- Extension of *A* and *B* must have something in common. (Filled area must contain at least one object)
- *A* and *B* can overlap, be equal, or be contained.
- Same situations make “No *A*s are *B*s” **false**.



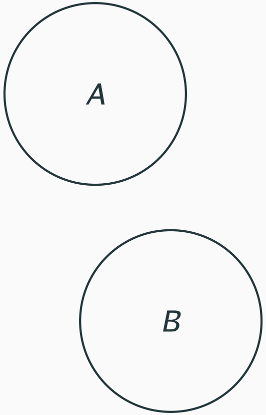
Making "Some *A*s are *B*s" true

- $(\exists x)(Ax \,\&\, Bx)$
- Extension of *A* and *B* must have something in common. (Filled area must contain at least one object)
- *A* and *B* can overlap, be equal, or be contained.
- Same situations make "No *A*s are *B*s" **false**.



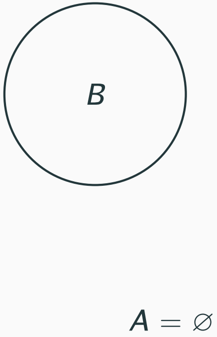
Making “Some *A*s are *B*s” false

- $\sim(\exists x)\,(Ax \,\&\, Bx)$
- Extension of *A* and *B* must have nothing in common.
- *A* and *B* don’t overlap, or one or both empty.
- Same situations make “No *A*s are *B*s” **true**.



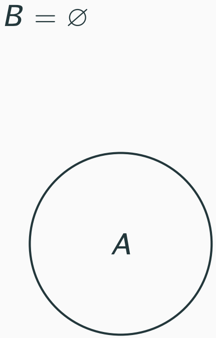
Making "Some As are Bs" false

- $\sim(\exists x)\,(Ax \,\&\, Bx)$
- Extension of A and B must have nothing in common.
- A and B don't overlap, or one or both empty.
- Same situations make "No As are Bs" **true**.



Making “Some *A*s are *B*s” false

- $\sim(\exists x)\,(Ax \,\&\, Bx)$
- Extension of *A* and *B* must have nothing in common.
- *A* and *B* don’t overlap, or one or both empty.
- Same situations make “No *A*s are *B*s” **true**.



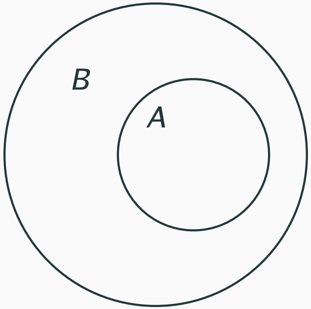
Making "Some *A*s are *B*s" false

- $\sim(\exists x)\,(Ax\ \&\ Bx)$
- Extension of *A* and *B* must have nothing in common.
- *A* and *B* don't overlap, or one or both empty.
- Same situations make "No *A*s are *B*s" **true**.

$$A = B = \varnothing$$

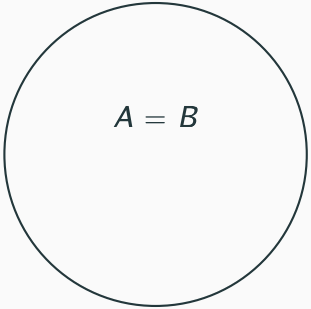
Making "All *A*s are *B*s" true

- $(\forall x)(Ax \supset Bx)$
- Extension of *A* must be contained in extension of *B*.
- Extensions of *A* and *B* can be the same.
- Extension of *A* can be empty.
- Same situations make . . .
  - "Only *B*s are *A*s" **true**.
  - "Some *A*s are not *B*s" **false**.



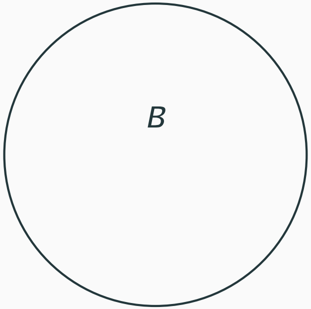
Making "All *A*s are *B*s" true

- $(\forall x)(Ax \supset Bx)$
- Extension of *A* must be contained in extension of *B*.
- Extensions of *A* and *B* can be the same.
- Extension of *A* can be empty.
- Same situations make ...
  - "Only *B*s are *A*s" **true**.
  - "Some *A*s are not *B*s" **false**.



Making "All *A*s are *B*s" true

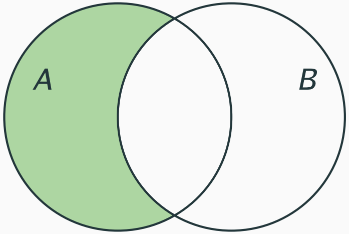
- $(\forall x)(Ax \supset Bx)$
- Extension of *A* must be contained in extension of *B*.
- Extensions of *A* and *B* can be the same.
- Extension of *A* can be empty.
- Same situations make ...
  - "Only *B*s are *A*s" **true**.
  - "Some *A*s are not *B*s" **false**.



$A = \varnothing$

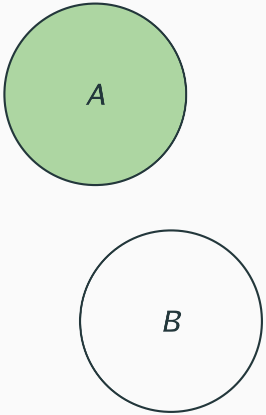
Making "All *A*s are *B*s" false

- $(\forall x)(Ax \supset Bx)$
- Extension of *A* must contain something not in *B*.
- Extensions of *A* cannot be empty, but *B* may be empty.
- Same situations make ...
  - "Only *B*s are *A*s" **false**.
  - "Some *A*s are not *B*s" **true**.



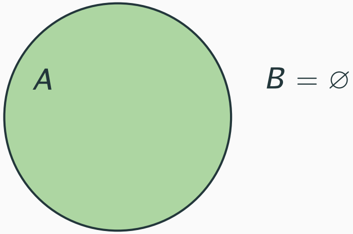
Making “All *A*s are *B*s” false

- $(\forall x)(Ax \supset Bx)$
- Extension of *A* must contain something not in *B*.
- Extensions of *A* cannot be empty, but *B* may be empty.
- Same situations make . . .
  - “Only *B*s are *A*s” **false**.
  - “Some *A*s are not *B*s” **true**.



Making “All *A*s are *B*s” false

- $(\forall x)(Ax \supset Bx)$
- Extension of *A* must contain something not in *B*.
- Extensions of *A* cannot be empty, but *B* may be empty.
- Same situations make . . .
  - “Only *B*s are *A*s” **false**.
  - “Some *A*s are not *B*s” **true**.



9. Semantics of QL

d. Testing for validity

Arguments involving quantifiers

1. If an action x is morally wrong then A is blameworthy for freely doing x.

2. If x is rationally optimal (there is no action which A has reason to think there is more reason for A to do), then A is not blameworthy for freely doing x.

3. Therefore, if x is morally wrong, then x is not rationally optimal. (Principle of moral categoricity.)

-John Skorupski, *Ethical Explorations*, 2000 (link)

Symbolizing Skorupski

1. If an action x is morally wrong then A is blameworthy for freely doing x.
2. If x is rationally optimal, then A is not blameworthy for freely doing x.
3. Therefore, if x is morally wrong, then x is not rationally optimal.

Domain:	actions
Wx:	x is morally wrong
Bx:	A is blameworthy for freely doing x
Ox:	x is rationally optimal

$(\forall x)(Wx \supset Bx)$

$(\forall x)(Ox \supset \sim Bx)$

$\therefore (\forall x)(Wx \supset \sim Ox)$

Symbolizing Skorupski

Domain:	actions
Wx:	x is morally wrong
Bx:	A is blameworthy for freely doing x
Ox:	x is rationally optimal

$(\forall x)(Wx \supset Bx)$

$(\forall x)(Ox \supset {\sim}Bx)$

$\therefore (\forall x)(Wx \supset {\sim}Ox)$

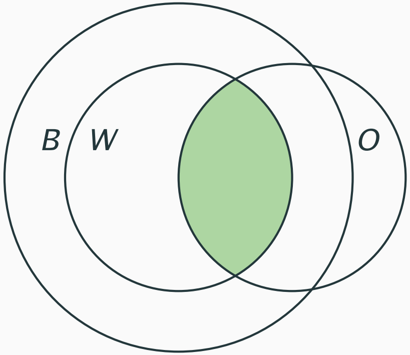
All Ws are Bs

No Os are Bs (iff No Bs are Os)

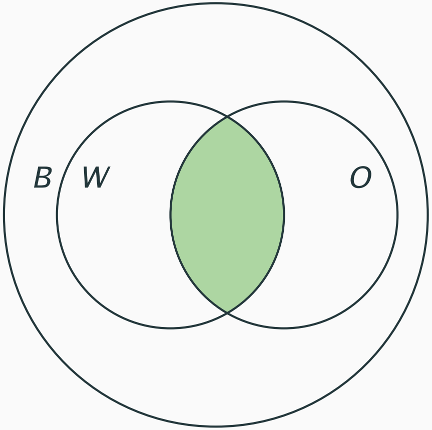
$\therefore$ No Ws are Os

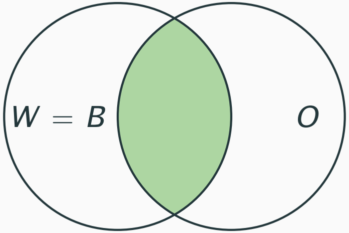
Determining validity

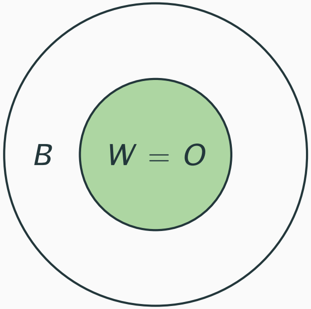
- Make conclusion $(\forall x)(Wx \supset \sim Ox)$ false.
- Make $(\exists x)(Wx \,\&\, Ox)$ true.
- Make $(\forall x)(Wx \supset Bx)$ true.
- $(\exists x)(Ox \,\&\, Bx)$ is now forced to be true.
- So, $(\forall x)(Ox \supset \sim Bx)$ is false.
- But those are not the only possibilities!



Other configurations







9. Semantics of QL

e. Semantic notions in QL

Semantics notions in QL

- $\mathcal{P}_1, \ldots, \mathcal{P}_n \vDash \mathcal{Q}$ if no interpretation makes all of $\mathcal{P}_1, \ldots, \mathcal{P}_n$ true and $\mathcal{Q}$ false.
- $\mathcal{P}$ is a **validity** ($\vDash \mathcal{P}$) if it is true in every interpretation.
- $\mathcal{P}$ and $\mathcal{Q}$ are **equivalent in QL** if no interpretation makes one true but the other false.
- $\mathcal{P}_1, \ldots, \mathcal{P}_n$ are **jointly satisfiable in QL** if some interpretation makes all of them true

Using interpretations

- By providing one suitable interpretation we **can** show that...
  - an argument is **not valid** in QL
  - a sentence is **not a validity** in QL
  - two sentences are **not equivalent** in QL
  - some sentences **are satisfiable** in QL
- But we **cannot** show using any number of interpretations that...
  - an argument **is valid** in QL
  - a sentence **is a validity** in QL
  - two sentences **are equivalent** in QL
  - some sentences **are not satisfiable** in QL

Examples

- $(\forall x)(Ax \vee Bx)$ and $(\forall x)\, Ax \vee (\forall x)\, Bx$ are not equivalent.
- $(\forall x)(Ax \supset Bx), (\forall x)(Ax \supset {\sim}Bx)$ are jointly satisfiable.
- $(\forall x)({\sim}Ax \supset Bx), (\exists x)(Bx \,\&\, Cxb) \nvDash (\exists x)({\sim}Ax \,\&\, Cxb)$.
- $\nvDash (\exists x)\, Aax \supset (\exists x)\, Axx$.

Test solutions on this week's practice problems!

9. Semantics of QL

f. Arguing about interpretations

Arguing about Interpretations

- No interpretation(s) can show that an argument is valid.
- That's because there is no way to inspect all possible interpretations.
- But we can show that arguments are valid, by:
  - a formal proof (a future topic!)
  - an informal argument
- The informal argument makes use of the **truth conditions** for sentences of QL.
- Analogous to arguing about valuations in SL.

Example

$$((\forall x)\mathcal{A}x \lor (\forall x)\mathcal{B}x) \vDash (\forall x)(\mathcal{A}x \lor \mathcal{B}x)$$

- Suppose an interpretation makes premise $(\forall x)\mathcal{A}x \lor (\forall x)\mathcal{B}x$ true.
- By truth conditions for $\lor$, it makes either $(\forall x)\mathcal{A}x$ or $(\forall x)\mathcal{B}x$ true.
- Suppose it's the first, i.e., $(\forall x)\mathcal{A}x$ is true.
  - By truth conditions for $\forall$, every object in the domain satisfies $\mathcal{A}x$.
  - By the truth conditions for $\lor$, every object satisfies $\mathcal{A}x \lor \mathcal{B}x$
  - So, by the truth conditions for $\forall$, $(\forall x)(\mathcal{A}x \lor \mathcal{B}x)$ is true.
- Suppose it's the second, i.e., $(\forall x)\mathcal{B}x$ is true: Similarly. . .
- These are the only possibilities: so any interpretation that makes the premise true must also make the conclusion true.

10. Proofs in QL

10. Proofs in QL

a. Some Equivalences in QL

Motivation: Proving Equivalences

- We'd like to have a method for proving that two sentences are equivalent, or that an argument is valid
- Models only provide counterexamples to equivalence or validity
- By extending our natural deduction system to cover QL, we'll be able to prove two sentences are equivalent!
  - Derive one from the other and vice versa
  - Basically: show that the biconditional of the two sentences is a tautology

Some Equivalences: Quantifiers commuting over connectives

1. $Fa \,\&\, (\exists x)Gx$ is equivalent to $(\exists x)(Fa \,\&\, Gx)$
2. $Fa \,\&\, (\forall x)Gx$ is equivalent to $(\forall x)(Fa \,\&\, Gx)$
3. $Fa \vee (\exists x)Gx$ is equivalent to $(\exists x)(Fa \vee Gx)$
4. $Fa \vee (\forall x)Gx$ is equivalent to $(\forall x)(Fa \vee Gx)$
5. $Fa \supset (\exists x)Gx$ is equivalent to $(\exists x)(Fa \supset Gx)$
6. $Fa \supset (\forall x)Gx$ is equivalent to $(\forall x)(Fa \supset Gx)$
   - But note that ‘$\supset$’ is not symmetric, so we have to examine the converse: e.g. $(\exists x)Gx \supset Fa$!
   - And there are no analogous rules for biconditionals

Quantifiers NOT commuting over some conditionals

'$\supset$' is not symmetric, so we have to examine the converse

bad: $(\exists x)Gx \supset Fa$ is NOT equivalent to $(\exists x)(Gx \supset Fa)$

Instead, the existential converts to a universal when we scope over the consequent:

7. $(\exists x)Gx \supset Fa$ is equivalent to $(\forall x)(Gx \supset Fa)$

bad: $(\forall x)Gx \supset Fa$ is NOT equivalent to $(\forall x)(Gx \supset Fa)$

Instead, the universal converts to an existential when we scope over the consequent:

8. $(\forall x)Gx \supset Fa$ is equivalent to $(\exists x)(Gx \supset Fa)$

Informal Argument: If anyone G's, then c F's

7. $(\exists x)Gx \supset Fc$ is equivalent to $(\forall x)(Gx \supset Fc)$

- Recall: "Any" in antecedents but **without** pronouns referring back to them are **existential**:

  If **anyone** is a hero, Greta is.

  Roughly: if there are heroes (at all), Greta is a hero.

  $(\exists x)\, Hx \supset Hg$

- Intuitively: for everybody, if they are a hero, then Greta is a hero

  $(\forall x)(Hx \supset Hg)$

- We would like to show that if we can derive one of these sentences, then we can derive the other.

  Our proof system will show this *naturally*!

Informal Argument: If everyone G's, then c F's

8. $(\forall x)Gx \supset Fc$ is equivalent to $(\exists x)(Gx \supset Fc)$

- By truth conditions for '$\supset$', first sentence is equivalent to:

  $\sim(\forall x)Gx \vee Fc$

  Apply de Morgan's: $(\exists x)\sim Gx \vee Fc$

- Now apply our rule 3: existential commutes over disjunction (provided the disjunct doesn't contain the bound variable x)

  $(\exists x)(\sim Gx \vee Fc)$, and apply '$\supset$' truth conditions again:

- Yields $(\exists x)(Gx \supset Fc)$

10. Proofs in QL

b. Rules for ∀

Rules for formal proofs

- Need rules for $\forall$ and $\exists$ for formal proofs
- Formal proofs now more important, because no alternative (truth–table method)
- Intro and Elim rules should be
  - simple
  - elegant (not involve other connectives or quantifiers)
  - yield only valid arguments

Candidates for rules

- Only simple sentence close to $(\forall x)\,\mathcal{A}\chi$ is $\mathcal{A}c$

- Gives simple, elegant $\forall$E rule:

$$\begin{array}{r|ll} k & (\forall \chi)\,\mathcal{A}\chi & \\ & & \\ & \mathcal{A}c & :k\ \forall\text{E} \end{array}$$

- This is a good rule: $(\forall \chi)\,\mathcal{A}\chi \models \mathcal{A}c$.

Candidates for rules

- Problem: corresponding "intro rule" isn't valid:

$$\begin{array}{r|ll} k & \mathcal{A}c & \\ & & \\ & (\forall \chi)\,\mathcal{A}\chi & :k \ \textbf{(doesn't follow)} \end{array}$$

- Diagnosis: the c in $\mathcal{A}c$ is a name for a **specific object**.
- We need a name for an **arbitrary, unspecified object**.
- If $\mathcal{A}c$ is true for whatever c **could** name, then $\mathcal{A}\chi$ is satisfied by **every** object.

Names for arbitrary objects

- When we give proofs of general claims, we often do use names for arbitrary objects (well, mathematicians do at least).

  All heroes admire Greta.

  Only people who wear capes admire Greta.

  ∴ All heroes wear capes.

  Proof: Let Carl be any hero.
  Since all heroes admire Greta, Carl admires Greta.
  Since only people who wear capes admire Greta, Carl wears a cape. But “Carl” stands for **any** hero.
  So all heroes wear capes.

Universal generalization

$$\begin{array}{r|ll} k & \mathcal{A}c & \\ & & \\ & (\forall \chi)\, \mathcal{A}\chi & :k\ \forall\mathrm{I} \end{array}$$

- c is special: c must not appear in any premise or assumption of a subproof not already ended
- $\mathcal{A}\chi$ is obtained from $\mathcal{A}c$ by replacing **all** occurrences of c by χ.
- In other words, c must also not occur in $\forall \chi\, \mathcal{A}\chi$.

General conditional proof

Proving “All *A*s are *B*s”

k	$\mid\ \mid\ Ac$	:AS for $\supset$I
l	$\mid\ \mid\ Bc$	
$l+1$	$\mid\ Ac \supset Bc$	:k-l $\supset$I
$l+2$	$\mid\ (\forall x)(Ax \supset Bx)$	:$l+1$ $\forall$I

Example

All heroes admire Greta.

Only people who wear capes admire Greta.

$\therefore$ All heroes wear capes.

$(\forall x)(Hx \supset Axg)$

$(\forall x)(Axg \supset Cx)$

$\therefore$ $(\forall x)(Hx \supset Cx)$

Let's do it on Carnap (PP10.2)!

Example

1	$(\forall x)(Hx \supset Axg)$	:PR
2	$(\forall x)(Axg \supset Cx)$	:PR
3	Hc	:AS for ⊃I
4	$Hc \supset Acg$	:1 ∀E
5	Acg	:4, 3 ⊃E
6	$Acg \supset Cc$	:2 ∀E
7	Cc	:6, 5 ⊃E
8	$Hc \supset Cc$	:3-7 ⊃I
9	$(\forall x)(Hx \supset Cx)$	:8 ∀I

Example

1	$(\forall x)\, Ax \vee (\forall x)\, Bx$	:PR
2	$\quad (\forall x)\, Ax$	:AS for $\vee$E
3	$\quad Ac$	:2 $\forall$E
4	$\quad Ac \vee Bc$	:3 $\vee$I
5	$\quad (\forall x)\, Bx$	:AS for $\vee$E
6	$\quad Bc$	:5 $\forall$E
7	$\quad Ac \vee Bc$	:6 $\vee$I
8	$Ac \vee Bc$	:1, 2–4, 5–7 $\vee$E
9	$(\forall x)(Ax \vee Bx)$	:8 $\forall$I

10. Proofs in QL

c. Rules for ∃

Rules for ∃

- If we know of a specific object that it satisfies $\mathcal{A}x$, we know that at least one object satisfies $\mathcal{A}x$.
- So this rule is valid:

$$
\begin{array}{l|ll}
k & \mathcal{A}c & \\
\hline
 & (\exists \chi)\,\mathcal{A}\chi & :k\ \exists\mathrm{I}
\end{array}
$$

Arbitrary objects again

- Problem: corresponding “elimination rule” isn’t valid:

$$\begin{array}{r|ll} k & (\exists x)\,\mathcal{A}x & \\ & & \\ & \mathcal{A}c & :k \ \textbf{doesn't follow from} \end{array}$$

- If we know that $(\exists x)\,\mathcal{A}x$ is true, we know that **at least one** object satisfies $\mathcal{A}x$, but not which one(s).
- To use this information, we have to introduce a temporary name that stands for any one of the objects that satisfy $\mathcal{A}(x)$.

Reasoning from existential information

- To use $(\exists \chi)\, \mathcal{A}\chi$, pretend the χ has a name c, and reason from $\mathcal{A}(c)$.
- This is what we do to reason informally from existential information, e.g.,

  There are heroes who wear capes.
  Anyone who wears a cape admires Greta.
  $\therefore$ Some heroes admire Greta.

  Proof: We know there are heroes who wear capes.
  Let Cate be an arbitrary one of them.
  So Cate wears a cape. Since anyone who wears a cape admires Greta, Cate admires Greta. Since Cate is a hero who admires Greta, some heroes admire Greta.

Existential elimination (mind the **restriction**!)

- If
  - we know that some object satisfies $\mathcal{A}\chi$,
  - we hypothetically assume that c is one of them (i.e., assume $\mathcal{A}c$),
  - and we can prove that $\mathcal{B}$ follows from this assumption,

  then $\mathcal{B}$ follows already from $(\exists\chi)\,\mathcal{A}\chi$.
- Rule for existential elimination:

$$
\begin{array}{l|l l}
k & (\exists\chi)\,\mathcal{A}\chi & \\
m & \quad \begin{array}{|l} \mathcal{A}c \\ \hline \end{array} & :\text{AS for } \exists\text{E} \\
n & \quad \begin{array}{|l} \mathcal{B} \end{array} & \\
 & \mathcal{B} & :k,\ m\text{-}n\ \exists\text{E}
\end{array}
$$

- c is special: c **must NOT appear outside subproof**

Example

There are heroes who wear capes.

Anyone who wears a cape admires Greta.

∴ Some heroes admire Greta.

$(\exists x)(Hx \,\&\, Cx)$

$(\forall x)(Cx \supset Axg)$

∴ $(\exists x)(Hx \,\&\, Axg)$

Example (PP10.5)

1	$(\exists x)(Hx \,\&\, Cx)$	:PR
2	$(\forall x)(Cx \supset Axg$	:PR
3	$\quad Hc \,\&\, Cc$	:AS for $\exists$E
4	$\quad Cc$	:3 & E
5	$\quad Cc \supset Acg$	:2 $\forall$E
6	$\quad Acg$	:4, 5 $\supset$E
7	$\quad Hc$	:3 & E
8	$\quad Hc \,\&\, Acg$	:4, 7 & I
9	$\quad (\exists x)(Hx \,\&\, Axg)$	:8 $\exists$I
10	$(\exists x)(Hx \,\&\, Axg)$	:1, 3-9 $\exists$I

10. Proofs in QL

d. The Rules, collected!

Rules for the Universal Quantifier

Universal Elimination (∀E)

$$\begin{array}{r|ll} m & (\forall \chi)\Phi(\ldots \chi \ldots \chi \ldots) & \\ \vdots & \quad \vdots & \\ s & \Phi(\ldots c \ldots c \ldots) & : m\ \forall\mathrm{E} \end{array}$$

- Note that you replace EVERY instance of χ with c

- Notation: $\Phi[c/\chi]$

- read "c for χ"

Universal Introduction (∀I)

$$\begin{array}{r|ll} m & \Phi(\ldots c \ldots c \ldots) & \\ \vdots & \quad \vdots & \\ s & (\forall \chi)\Phi(\ldots \chi \ldots \chi \ldots) & : m\ \forall\mathrm{I} \end{array}$$

Provided that both
(i) c does not occur in any undischarged assumptions that Φ is in the scope of.
(ii) χ does not occur already in $\Phi(\ldots c \ldots c \ldots)$.

Rules for the Existential Quantifier

Existential Introduction (∃I)

m	$\Phi(\ldots c \ldots c \ldots)$	
$\vdots$	$\vdots$	
s	$(\exists \chi)\Phi(\ldots \chi \ldots \chi \ldots)$	:m ∃I

- **Provided that** χ does not occur already in $\Phi(\ldots c \ldots c \ldots)$.

- Note that χ may replace some or all occurrences of c.

Existential Elimination (∃E)

m	$(\exists \chi)\Phi$	
	$\vdots$	
n	$\quad\vert\ \Phi(\ldots c \ldots c \ldots)$	:AS for ∃E
	$\quad\vert\ \vdots$	
s	$\quad\vert\ \Psi$	
$s+1$	Ψ	:m, n–s ∃E

provided that c doesn't occur **anywhere else outside** the subproof

10. Proofs in QL

e. Substitution Instances

(Full) Substitution Instances

- $\Phi[c/\chi]$ is the sentence you get from $(\forall\chi)\Phi$ by dropping the $(\forall\chi)$ quantifier and putting c in place of **every** χ in Φ.
- The other variables are untouched!
- $\Phi[c/\chi]$ can also arise from $(\exists\chi)\Phi$ by dropping the $(\exists\chi)$ and putting c in place of **every** χ in Φ.
- Equivalent notation: $\Phi\boxed{\chi \Rightarrow c}$

Some Examples of Substitution Instances

- Instances of $(\forall y)$Hy:
  - Ha, Hb, Hm_{11}
- Instances of $(\exists z)$Haz:
  - Haa, Hab, Haj_3
- Instances of $(\exists z)(Hz \,\&\, Fzz)$:
  - Remember to replace **EVERY** occurance of z with the chosen constant:
  - $(Ha \,\&\, Faa)$, $(Hc \,\&\, Fcc)$
  - The following are **NOT** substitution instances:
  - $(Ha \,\&\, Faz)$, $(Hy \,\&\, Faa)$, $(Ha \,\&\, Fab)$

Some Things to WATCH OUT for!

- Is the wff ‘$(\forall x)Gx \vee (\forall x)Fx$’ universally quantified?
- NO! It is a disjunction, since ‘$\vee$’ is its main connective.
- What are instances of a sentence like $(\forall x)Gx \vee (\forall x)Fx$?
- Trick question! It has no instances, since it is a disjunction, not a quantified sentence!

Finding Formulas that have a given instance

- Goal: figure out some wffs that a given wff is an instance of:
- Fa is an instance of...
  - $(\forall x)Fx$, $(\exists x)Fx$, $(\forall z)Fz$, etc.
- Fab is an instance of...
  - $(\forall x)Fax$, $(\forall x)Fxb$, $(\exists x)Fax$, $(\exists x)Fxb$
- Faa is an instance of...
  - $(\forall x)Fxx$, $(\forall x)Fax$, $(\forall x)Fxa$, $(\exists x)Fxx$, $(\exists x)Fxa$

Partial Substitution Instances

- For Existential Introduction, we can use a partial substitution instance of the wff Φ:
- '$\Phi\lceil \chi / c \rceil$' indicates that the variable χ does not need to replace all occurrences of the constant c in Φ.
- You can decide which occurrences of c to replace and which to leave in place

Examples of Partial Substitution Instances!

- ‘$\Phi\lceil\chi/c\rceil$’ indicates that the variable χ does not need to replace all occurrences of the constant c in Φ

1	Rdd	
2	$(\exists x)Rxx$	:1 ∃I
3	$(\exists x)Rxd$	:1 ∃I
4	$(\exists z)Rdz$	:1 ∃I
5	$(\exists y)(\exists z)Ryz$	:4 ∃I

Existential Introduction (∃I)

m	$\Phi(c)$	
$\vdots$	$\vdots$	
s	$(\exists\chi)\Phi\lceil\chi/c\rceil$	:m ∃I

– **Provided that** χ does not occur already in $\Phi(\ldots c \ldots c \ldots)$.

Putting this Notation to Work!

- Using the notation of (partial) substitution instances, we can rewrite our rule schemata!
- Payoff: we can drop one of the notes we had about existential introduction
- Is this worth it???? Only one way to find out!

Rules for the Universal Quantifier

Universal Elimination (∀E)

$$\begin{array}{l|ll}
m & (\forall \chi)\Phi & \\
\vdots & \quad\vdots & \\
s & \Phi[c/\chi] & :m\ \forall\text{E}
\end{array}$$

- Note that you replace **EVERY** instance of χ with c

- Notation: $\Phi[c/\chi]$

- read "c for χ"

Universal Introduction (∀I)

$$\begin{array}{l|ll}
m & \Phi(c) & \\
\vdots & \quad\vdots & \\
s & (\forall \chi)\Phi[\chi/c] & :m\ \forall\text{I}
\end{array}$$

Provided that both
(i) c does not occur in any undischarged assumptions that Φ is in the scope of.
(ii) χ does not occur already in $\Phi(\ldots c \ldots c \ldots)$.

Rules for the Existential Quantifier

Existential Introduction (∃I)

$$\begin{array}{l|ll}
m & \Phi(c) & \\
\vdots & \quad\vdots & \\
s & (\exists\chi)\Phi\ulcorner\chi/c\urcorner & :m\ \exists\mathrm{I}
\end{array}$$

- **Provided that** χ does not occur already in $\Phi(\ldots c \ldots c \ldots)$.
- As indicated by $\ulcorner\chi/c\urcorner$, χ **may** replace **just some** occurrences of c

Existential Elimination (∃E)

$$\begin{array}{l|l|ll}
m & \multicolumn{2}{l}{(\exists\chi)\Phi} & \\
 & \multicolumn{2}{l}{\quad\vdots} & \\
n & & \Phi[c/\chi] & :\text{AS for } \exists\text{E} \\
\cline{3-3}
 & & \quad\vdots & \\
s & & \Psi & \\
s+1 & \multicolumn{2}{l}{\Psi} & :m,\ n\text{-}s\ \exists\text{E}
\end{array}$$

provided that c doesn’t occur **anywhere else outside** the subproof

10. Proofs in QL

f. Tips for an all-natural look!

Working Backwards

- Work backwards (at least in making a plan):
- If the final sentence is...
  - existentially quantified, you will probably introduce it with $\exists I$ somewhere
  - universally quantified: will probably use $\forall I$
  - a conditional: conditional introduction
  - a disjunction: disjunction intro

Working with your Premises

- If a premise is. . .
  - universally quantified: you will probably eliminate that universal at least once to arrive at an instance
    (you may do this multiple times for different instances)
  - existentially quantified: you will probably eliminate this existential, and if so you probably want to start this subproof ASAP (mind the tricky syntax!)
  - a conjunction: you can get conjuncts using &E
  - a disjunction: think about using disjunction elimination
    (mind the tricky syntax!)

Advice for deriving $(\exists \chi)\, \mathcal{A}\chi$, e.g. $(\exists y)\, Py$

- Ask whether you can derive an instance $\mathcal{A}$[c for χ], e.g. Pc
- If so, you could introduce $(\exists \chi)\, \mathcal{A}\chi$ through $\exists I$
  - Remember: you might have to do this first as the last line of a SUBproof started for $\exists E$, before repeating what you want below the subproof, justified by $\exists$ ELIMINATION
- If not: consider using negation–elimination or disjunction elimination (if a disjunction is available)

Advice for using Existential Elimination

- Remember to begin a sub-proof with a relevant instance of the existential
  - hot tip: use a constant that hasn't appeared yet!
  - in *Carnap*, tab in to a new proof-level by pressing 'tab'

- **PRO TIP**: Usually a good idea to carry out *as much of your proof as possible* **within the scope** of the subproof: this makes obeying the restrictions easier
- Remember that you'll write the same sentence TWICE in a row, once within the subproof and once outside (the second instance being the one justified by $\exists E$)
- Remember that the constant c must NOT appear in the conclusion, existential being eliminated, or an undischarged assumption

11. Multiple quantifiers & The Identity Predicate

11. Multiple quantifiers & The Identity Predicate

a. Two quantifiers

Formulas expressing relations

- A formula $\mathcal{A}x$ with one free variable expresses a **property**
- A formula $\mathcal{B}xy$ with two free variables expresses a **relation**
- $(\forall x)(\forall y)\,\mathcal{B}xy$ is a sentence:
- It's true iff **every pair** of objects α, β stand in the relation expressed by $\mathcal{B}xy$.
- $(\exists x)(\exists y)\,\mathcal{B}xy$ is a sentence:
- It's true iff **at least one pair** of objects α, β stand in the relation expressed by $\mathcal{B}xy$.

Multiple uses of a single quantifier: $\forall$

- Axy: x admires y.
- $(\forall x)(\forall y)\, Axy$: for every pair $\langle \alpha, \beta \rangle$, α admires β.
- In other words: everyone admires everyone.
- Note: "every pair" includes pairs $\langle \alpha, \alpha \rangle$, i.e.,
- $(\forall x)(\forall y)\, Axy$ is true only if all pairs $\langle \alpha, \alpha \rangle$ satisfy Axy.
- That means, everyone admires themselves, in addition to everyone else.
- So: $(\forall x)(\forall y)\, Axy$ does **not** symbolize "everyone admires everyone **else**." (To handle that, we'll need identity!)

Multiple uses of single quantifier: $\exists$

- $(\exists x)(\exists y)\, Axy$: for at least one pair $\langle \alpha, \beta \rangle$, α admires β.
- In other words: at least one person admires at least one person.
- Note: includes pairs $\langle \alpha, \alpha \rangle$, i.e.,
- $(\exists x)(\exists y)\, Axy$ is already true if a single pair $\langle \alpha, \alpha \rangle$ satisfies Axy.
- That means, we could just have one person admiring themselves.
- So: $(\exists x)(\exists y)\, Axy$ does **not** symbolize "someone admires someone **else**." (again, for that, we'll need the identity predicate)

Alternating quantifiers

1. $(\forall x)(\exists y)\, Axy$
   Everyone admires someone
   (possibly themselves)
2. $(\forall y)(\exists x)\, Axy$
   Everyone is admired by someone
   (not necessarily the same person)
3. $(\exists x)(\forall y)\, Axy$
   Someone admires everyone
   (including themselves)
4. $(\exists y)(\forall x)\, Axy$
   Someone is admired by everyone
   (including themselves)

Convergence vs. uniform convergence

- A function f is **point-wise continuous** if

$$(\forall \epsilon)(\forall x)(\forall y)(\exists \delta)(|x - y| < \delta \supset |f(x) - f(y)| < \epsilon)$$

- A function f is **uniformly continuous** if

$$(\forall \epsilon)(\exists \delta)(\forall x)(\forall y)(|x - y| < \delta \supset |f(x) - f(y)| < \epsilon)$$

11. Multiple quantifiers & The Identity Predicate

b. Multiple Determiners

“Determiner phrases” say what?

- Determiners: quantifiers and indefinite or definite articles (also possessives and demonstratives)
- e.g. many, some, a, the, his, their, this, that
- Determiner phrases: combine a determiner with a (possibily modified) noun:
- ‘all heroes’; ‘a cape’
- ‘some woman’; ‘the donkey’

Symbolizing multiple determiners

- What if your sentence contains more than one determiner phrase?
- Deal with each determiner separately!
- Think of determiner phrase as replaced with name or variable—result has one less determiner.
- When you're down to one determiner, apply known methods for single quantifiers.
- This results in formulas that express properties or relations, but themselves contain quantifiers.

Two separate determiner phrases

- All heroes wear a cape
- All heroes satisfy "x wears a cape"

$$(\forall x)(Hx \supset \text{“}x \text{ wears a cape”})$$

- x wears a cape

$$(\exists y)(Ey \,\&\, Rxy)$$

- Together:

$$(\forall x)(Hx \supset (\exists y)(Ey \,\&\, Rxy))$$

Determiner within determiner phrase

- All heroes who wear a cape admire Greta.
- All things that satisfy "*x* is a hero who wears a cape" admire Greta.

$$(\forall x)(\text{“x is a hero who wears a cape”} \supset Axg)$$

- *x* is a hero who wears a cape

$$Hx \,\&\, (\exists y)(Ey \,\&\, Rxy)$$

- Together:

$$(\forall x)((Hx \,\&\, (\exists y)(Ey \,\&\, Rxy)) \supset Axg)$$

“Any” is sometimes existential

- Any (every) cape is worn by a hero:

$$(\forall x)(Ex \supset (\exists y)(Hy \,\&\, Ryx))$$

- No hero wears any cape:

$$(\forall x)(Hx \supset \sim(\exists y)(Ey \,\&\, Rxy))$$
$$\sim(\exists x)(Hx \,\&\, (\exists y)(Ey \,\&\, Rxy))$$

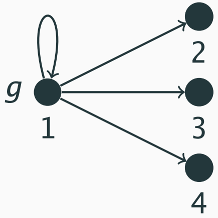
- No hero wears every cape:

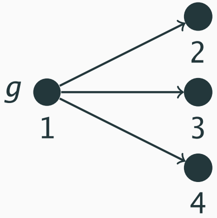
$$(\forall x)(Hx \supset \sim(\forall y)(Ey \supset Rxy))$$
$$\sim(\exists x)(Hx \,\&\, (\forall y)(Ey \supset Rxy))$$

11. Multiple quantifiers & The Identity Predicate

c. The identity predicate

Greta admires everyone (else)





Greta admires everyone.

$(\forall x)\ Agx$

Greta admires everyone **else**.

$(\forall x)(\text{“}x\text{ is not Greta”} \supset Agx)$

$(\forall x)(\sim x = g \supset Agx)$

The identity predicate

- A new, special two-place predicate: $=$
  - Written between arguments, **without parentheses**.
  - Needs no mention in symbolization key.
  - Always interpreted the same: extension of '$=$' is all pairs $\langle\alpha, \alpha\rangle$.
- '$a = b$' true iff 'a' and 'b' name one and the same object.
- $x = y$ satisfied by all and only the pairs $\langle\alpha, \alpha\rangle$.
- $\sim x = y$ is satisfied by a pair $\langle\alpha, \beta\rangle$ iff α and β are different objects.

MISTAKES! Ungrammatical expressions with identity

- $x = {\sim}y$ is not grammatical.

  $\sim$ can only go in front of a formula, and y is not one.

- ${\sim}(x = y)$ is also not grammatical.

  ‘$(x = y)$’ is also not a formula.

- *Carnap* will not tolerate this nonsense! Take heed!

‘Something else’ and ‘everything else’

- Remember: different variables do NOT entail different objects.
- $(\exists x)(\exists y)\,Axy$ doesn’t mean that someone admires someone else.
- It just means that someone admires someone (possibly themselves).
- To symbolize “someone else” add $\sim x{=}y$:

$$(\exists x)(\exists y)(\sim x{=}y \,\&\, Axy)$$

- $(\forall x)(\forall y)\,Axy$ says that everyone admires everyone (including themselves).
- To symbolize “everyone admires everyone else” add $\sim x{=}y$:

$$(\forall x)(\forall y)(\sim x{=}y \supset Axy)$$

‘Something else’ and ‘everything else’

- The closest quantifier (typically) determines whether you should use & or $\supset$:

  $(\forall x)(\exists y)(\sim x{=}y \,\&\, Axy)$ vs. $(\exists x)(\forall y)(\sim x{=}y \supset Axy)$

  Everyone admires someone else vs. Someone admires everyone else

- If you have mixed domains, it works the same way:
- Recall predicate ‘Px’: “x is a person”
- Everyone admires someone **else**:

$$(\forall x)(Px \supset (\exists y)((Py \,\&\, \sim x{=}y) \,\&\, Axy))$$

- Someone admires everyone **else**:

$$(\exists x)(Px \,\&\, (\forall y)((Py \,\&\, \sim x{=}y) \supset Axy)$$

Other than, except

- "**Someone other than Greta** is a hero":

  $(\exists x)(\sim x = g \,\&\, Hx)$

- "**Everyone other than Greta** is a hero"; same as:
- "**Everyone except Greta** is a hero":

  $(\forall x)(\sim x = g \supset Hx)$

‘No-one other than’ vs. Singular “only”

- “**No-one other than Greta** is a hero”:

  $\sim(\exists x)(Hx \mathbin{\&} \sim x{=}g)$

  $(\forall x)(Hx \supset x{=}g)$

- “**Only Greta** is a hero”:
- Content: No-one other than Greta is a hero, **AND** Greta is a hero:

  $(\forall x)(Hx \supset x{=}g) \mathbin{\&} Hg$

  $(\forall x)(Hx \equiv x{=}g)$

Uniqueness

- Non–unique: "There is at least one hero":

$$(\exists x)\, Hx$$

- Unique: "There is exactly one hero":
  - There's at least one hero, AND
  - There are no others:

$$(\exists x)\,(Hx \mathbin{\&} \sim(\exists y)\,(\sim y = x \mathbin{\&} Hy))$$
$$(\exists x)\,(Hx \mathbin{\&} (\forall y)(Hy \supset x{=}y))$$

  - Or more succinctly: $(\exists x)(\forall y)(Hy \equiv x{=}y)$

11. Multiple quantifiers & The Identity Predicate

d. Numerical quantification

Numerical Quantification: *n*-many as ‘at least *n*’

- Cardinal numbers can be determiners:
  - **Three heroes** wear capes.
- Not always clear if “three heroes” means exactly vs. **at least** three hero
- We’ll assume the latter.
  - Do you have two dollars? Yes, I have two dollars. (Uncontroversially true even if you have more than $2)
- Using QL, we can express the following kinds of sentences:
  - **At least** *n* people are . . .
  - **Exactly** *n* people are . . .
  - **At most** *n* people are . . .
- i.e. we can count on QL!

At least n

- At least 1 hero is inspiring:

$$(\exists x)(Hx \,\&\, Ix)$$

- At least 2 heroes are inspiring:

$$(\exists x)(\exists y)(\sim x{=}y \,\&\, ((Hx \,\&\, Ix) \,\&\, (Hy \,\&\, Iy)))$$

- At least 3 heroes are inspiring:

$$(\exists x)(\exists y)(\exists z)\Big((\sim x{=}y \,\&\, (\sim y = z \,\&\, \sim x = z)) \,\&\, ((Hx \,\&\, Ix) \,\&\, ((Hy \,\&\, Iy) \,\&\, (Hz \,\&\, Iz)))\Big)$$

At least n

- There are at least n As, i.e. "$(\exists^{\geq n} x)\, Ax$":

$$\begin{aligned}(\exists x_1) \ldots (\exists x_n)\Big((\sim x_1 = x_2 \,\&\, (\sim x_1 = x_3 \,\&\, \ldots \,\&\, (\sim x_1 = x_n \,\&\, \\ (\sim x_2 = x_3 \,\&\, \ldots \,\&\, (\sim x_2 = x_n \,\&\, \\ \ddots \\ \sim x_{n-1} = x_n) \ldots) \,\&\, \\ (Ax_1 \,\&\, (Ax_2 \,\&\, \ldots \,\&\, Ax_n) \ldots)\Big)\end{aligned}$$

At least *n*

- Note: must state that **every pair** of variables is different, e.g.,

$$(\exists x_1)(\exists x_2)(\exists x_3)((\sim x_1 = x_2 \;\&\; \sim x_2 = x_3) \;\&\; (Hx_1 \;\&\; (Hx_2 \;\&\; Hx_3)))$$

  only says "There are at least two heroes"!
  - Take extension of Hx to be: 1, 2
  - Then 1 can play role of x_1 and x_3, 2 role of x_2.
  - Both "$\sim 1 = 2$" and "$\sim 2 = 3$" are true.
- At least *n* *B*s are *C*s: substitute '$Bx \;\&\; Cx$' for 'Ax':

$$(\exists^{\geq n} x)(Bx \;\&\; Cx)$$

Exactly one (i.e. Uniqueness; see above)

- There is exactly one hero:

$$(\exists x)(Hx \,\&\, {\sim}(\exists y)(Hy \,\&\, {\sim}x{=}y))$$

- This is equivalent to:

$$(\exists x)(Hx \,\&\, (\forall y)(Hy \supset x{=}y))$$

- In general: "g has property A **uniquely**":

$$Ag \,\&\, (\forall y)(Ay \supset g{=}y)$$

or just: $(\forall y)(Ay \equiv g{=}y)$

Exactly *n*

- There are exactly *n* *A*s, i.e. "$(\exists^{=n}x)\,Ax$":

$$\begin{aligned}(\exists x_1)\ldots(\exists x_n)\Big((\sim x_1 = x_2 \,\&\, (\sim x_1 = x_3 \,\&\, \ldots \,\&\, (\sim x_1 = x_n \,\&\\ (\sim x_2 = x_3 \,\&\, \ldots \,\&\, (\sim x_2 = x_n \,\&\\ \ddots\\ \sim x_{n-1} = x_n)\ldots)\,\&\\ (Ax_1 \,\&\, (Ax_2 \,\&\, \ldots \,\&\, Ax_n)\ldots))\,\&\\ (\forall y)(Ay \supset (y = x_1 \vee \cdots \vee y = x_n))\Big)\end{aligned}$$

- Exactly *n* *B*s are *C*s:

$$(\exists^{=n}x)(Bx \,\&\, Cx)$$

Exactly *n*

- There are exactly *n* *A*s, i.e. "$(\exists^{=n} x)\ Ax$":

$$(\exists x_1) \ldots (\exists x_n)\Big((\sim x_1 = x_2 \mathbin{\&} (\sim x_1 = x_3 \mathbin{\&} \ldots \mathbin{\&} (\sim x_1 = x_n \mathbin{\&}$$
$$(\sim x_2 = x_3 \mathbin{\&} \ldots \mathbin{\&} (\sim x_2 = x_n \mathbin{\&}$$
$$\ddots$$
$$\sim x_{n-1} = x_n) \ldots) \mathbin{\&}$$
$$(\forall y)(Ay \equiv (y = x_1 \vee \cdots \vee y = x_n))\Big)$$

- Exactly *n* *B*s are *C*s:

$$(\exists^{=n} x)(Bx \mathbin{\&} Cx)$$

At most n

- There are **at most n** As $\Leftrightarrow$ There are **not at least $n+1$** As

$$(\exists^{\leq n} x)\, Ax \Leftrightarrow \sim(\exists^{\geq (n+1)} x)\, Ax$$

- For instance: There are at most two heroes:

$$\sim(\exists x)(\exists y)(\exists z)((Hx \,\&\, (Hy \,\&\, Hz)) \,\&\, (\sim x = y \,\&\, (\sim x = z \,\&\, \sim y = z)))$$

$$(\forall x)(\forall y)(\forall z)((Hx \,\&\, (Hy \,\&\, Hz)) \supset (x = y \vee (x = z \vee y = z)))$$

- $\sim(\exists^{\geq (n+1)} x)\, Ax$ is equivalent to:

$$\begin{aligned} &(\forall x_1) \ldots (\forall x_{n+1})((Ax_1 \,\&\, \ldots \,\&\, Ax_{n+1}) \supset \\ &\quad (x_1 = x_2 \vee (x_1 = x_3 \vee \cdots \vee (x_1 = x_{n+1} \vee \\ &\quad\quad (x_2 = x_3 \vee \cdots \vee (x_2 = x_{n+1} \vee \\ &\quad\quad\quad \ddots \\ &\quad\quad\quad x_n = x_{n+1}) \ldots))) \end{aligned}$$

11. Multiple quantifiers & The Identity Predicate

e. Both ‘both’ and ‘neither’

Schematizing ‘Both’

- “Both heroes inspire”: this means that
  There are **exactly 2** heroes, and both inspire:

$$(\exists x)(\exists y)\Big(((\sim x{=}y \;\&\; (Hx \;\&\; Hy)) \;\& \\ (\forall z)(Hz \supset (z = x \vee z = y))) \;\& \\ (Ix \;\&\; Iy)\Big)$$

- Note: “Both heroes inspire” implies “There are exactly two inspiring heroes”, but not vice versa!
- e.g. if there are exactly two inspiring heros and one (or more) not–inspiring hero(s)

Schematizing ‘Neither’

- “Neither hero inspires”: this means that

  There are **exactly 2** heroes, and neither of them inspires:

$$(\exists x)(\exists y)\Big(((\sim x{=}y \;\&\; (Hx \;\&\; Hy)) \;\&$$
$$(\forall z)(Hz \supset (z = x \vee z = y))) \;\&$$
$$(\sim Ix \;\&\; \sim Iy)\Big)$$

11. Multiple quantifiers & The Identity Predicate

f. ‘The’ Definite Description

Definite descriptions

- Definite description: **the so-and-so**
- Russell's analysis of definite description: to say

  "The A is B"

  is to say:
- There is something, which:
  - is A,
  - is the **only** A (i.e. the unique thing that is A),
  - is B.
- In QL:

$$(\exists x)(Ax \,\&\, (\forall y)(Ay \supset x{=}y) \,\&\, Bx)$$

- or more succinctly:

$$(\exists x)(\forall y)\big((Ay \equiv x{=}y) \,\&\, Bx\big)$$

Example: ‘The author of Waverley is blah’

- Schematize “The author of *Waverley* is Scottish”:
- Use the following symbolization key:
- Ax: x is an author; Wxz: x wrote z; Sx: x is Scottish; ℓ: *Waverley*

$$(\exists x)(Ax \,\&\, Wx\ell \,\&\, (\forall y)((Ay \,\&\, Wy\ell) \supset x{=}y) \,\&\, Sx)$$

“The” vs. “exactly one”

- Compare:
  1. The hero inspires:

$$(\exists x)(Hx \mathbin{\&} (\forall y)(Hy \supset x{=}y) \mathbin{\&} Ix)$$

  2. There is exactly one inspiring hero:

$$(\exists x)(Hx \mathbin{\&} Ix \mathbin{\&} (\forall y)((Hy \mathbin{\&} Iy) \supset x{=}y))$$

- (2) can be true without (1), but not vice versa.
- (Namely when there is exactly one inspiring hero, but also a non-inspiring hero.)
- So (1) entails (2), but not vice versa.

Strawson's analysis (presuppositional theories)

- According to Russell, "The hero wears a cape" is **false** if there is no hero, or if there is more than one.
- Consider: "the present King of France is bald."
- P. F. Strawson disagrees with these truth conditions. Rather, we only succeed in making a statement if there is a unique hero (or a unique king of France).
- "There is a unique hero" is not part of what is **said** by a definite description, but is only presupposed.

Singular possessive (a definite description)

- Singular possessives form noun phrases, e.g., "Joe's cape"
- They work like definite descriptions:
  Joe's cape is the cape Joe owns. E.g.:
  - "Autumn wears Joe's cape" symbolizes the same as:
    "Autumn wears the cape that Joe owns":

$$(\exists x)\Big(\big((Ex \,\&\, Ojx) \,\&\, (\forall y)((Ey \,\&\, Ojy) \supset x{=}y)\big) \,\&\, Wax\Big)$$

Singular vs. plural possessive

- Compare **plural** possessives: those are '∀'s':
  - "Autumn wears Joe's cape**s**" symbolizes the same as:

    "Autumn wears every cape that Joe owns":

$$(\forall x)((Ex \,\&\, Ojx) \supset Wax)$$

- So plural possessives are NOT definite descriptions.

11. Multiple quantifiers & The Identity Predicate

g. Using quantifiers to express properties

Our symbolization key

Domain:	people alive in 2022 and items of clothing
a:	Autumn
g:	Greta
Px:	$\underline{\qquad}_x$ is a person
Lx:	$\underline{\qquad}_x$ is an item of clothing.
Ex:	$\underline{\qquad}_x$ is a cape
Rxy:	$\underline{\qquad}_x$ wears $\underline{\qquad}_y$
Hx:	$\underline{\qquad}_x$ is a hero
Ix:	$\underline{\qquad}_x$ inspires
Yxy:	$\underline{\qquad}_x$ is younger than $\underline{\qquad}_y$
Axy:	$\underline{\qquad}_x$ admires $\underline{\qquad}_y$
Oxy:	$\underline{\qquad}_x$ owns $\underline{\qquad}_y$

Expressing properties, revisited

- One-place predicates express properties, e.g.,

  Hx expresses property “being a hero”

- Combinations of predicates (with connectives, names) can express derived properties, e.g.,

  Axg expresses “x admires Greta”

  $Hx \,\&\, Cx$ expresses “x is a hero who wears a cape”

- Using quantifiers, we can express even more complex properties, e.g.,

  $(\exists y)(Py \,\&\, Axy)$ expresses “x admires someone”

Finding, using properties expressed

- If you can say it for Greta, you can say it for *x*.
  - Greta admires a hero.
    $(\exists y)(Hy \,\&\, Agy)$
  - *x* admires a hero.
    $(\exists y)(Hy \,\&\, Axy)$
- If you can say it for *x*, you can say it for Greta.
  - *x* wears a cape.
    $(\exists y)(Ey \,\&\, Rxy)$
  - Greta wears a cape.
    $(\exists y)(Ey \,\&\, Rgy)$

Ex: $\_\_\_\_\__x$ is a cape
Rxy: $\_\_\_\_\__x$ wears $\_\_\_\_\__y$

Examples

- x wears a cape.
  $(\exists y)(Ey \mathbin{\&} Rxy)$
- x is admired by everyone.
  $(\forall y)(Py \supset Ayx)$
- x admires a hero.
  $(\exists y)(Hy \mathbin{\&} Axy)$
- x admires only heroes.
  $(\forall y)(Axy \supset Hy)$
- x is unclothed (i.e. naked).
  $\sim(\exists y)(Ly \mathbin{\&} Rxy)$
  $(\forall y)(Ly \supset \sim Rxy)$

Px	$\_\_\_\__x$ is a person	Lx	$\_\_\_\__x$ is an item of clothing
Ex	$\_\_\_\__x$ is a cape	Rxy	$\_\_\_\__x$ wears $\_\_\_\__y$

11. Multiple quantifiers & The Identity Predicate

h. Multiple determiners: worked example

Mary Astell, 1666–1731

- British political philosopher
- *Some Reflections upon Marriage* (1700)
- In preface to 3rd ed. 1706 reacts to William Nicholls' claim (in *The Duty of Inferiors towards their Superiors, in Five Practical Discourses* (London 1701), Discourse IV: The Duty of Wives to their Husbands), that women are naturally inferior to men.

Astell TL;DR

- What can Nicholls possibly mean by “women are naturally inferior to men”?
- It can’t be that some woman is inferior to some man, since that’s “no great discovery.”
- After all, surely some men are inferior to some women.
- The obviously intended meaning must be: **all** women are inferior to **all** men.
- But that can’t be right, for then “the greatest Queen ought not to command but to obey her Footman.”
- It can’t even be just: **all** women are inferior to **some** men.
- Since “had they been pleased to remember their Oaths of Allegiance and Supremacy, they might have known that *One* Woman is superior to *All* the Men in these Nations.”

Symbolizing Astell

- Some woman is superior to every man
- Some woman satisfies “x is superior to every man”

$$(\exists x)(Wx \,\&\, \text{“}x \text{ is superior to every man”})$$

- x is superior to every man

$$(\forall y)(My \supset Sxy)$$

- Together:

$$(\exists x)(Wx \,\&\, (\forall y)(My \supset Sxy))$$

Formalizing Astell

- Some woman is superior to some man.

  $(\exists x)(Wx \,\&\, (\exists y)(My \,\&\, Sxy))$

- Every woman is superior to every man.

  $(\forall x)(Wx \supset (\forall y)(My \supset Sxy))$

- Every woman is superior to some man.

  $(\forall x)(Wx \supset (\exists y)(My \,\&\, Sxy))$

- Some woman is superior to every man.

  $(\exists x)(Wx \,\&\, (\forall y)(My \supset Sxy))$

11. Multiple quantifiers & The Identity Predicate

i. Quantifier scope ambiguity

More scope ambiguity

- “Autumn and Greta admire Isra or Luisa.”
- Two logically distinct, natural readings:

1) Autumn admires Isra or Luisa, **and** so does Greta.

$$(Aai \lor Aal) \mathbin{\&} (Agi \lor Agl)$$

2) Autumn and Greta both admire Isra, or they both admire Luisa.

$$(Aai \mathbin{\&} Agi) \lor (Aal \mathbin{\&} Agl)$$

Negation and the quantifiers

- "All heroes don't inspire"
  - Denial of "all heroes inspire". Ask: "Do all heroes inspire (Answer: No, *it's not the case that* all heroes inspire ")

$$\sim(\forall x)(Hx \supset Ix)$$
$$(\exists x)(Hx \,\&\, \sim Ix)$$

  - All heroes are not inspiring, i.e., No heroes inspire

$$(\forall x)(Hx \supset \sim Ix)$$
$$\sim(\exists x)(Hx \,\&\, Ix)$$

Multiple quantifiers and ambiguity

- "All heroes wear a cape"
    - "A cape" in the scope of "all heroes", i.e., "For every hero, there is a cape they wear"

$$(\forall x)(Hx \supset (\exists y)(Ey \mathbin{\&} Rxy))$$

$$(\forall x)(\exists y)(Hx \supset (Ey \mathbin{\&} Rxy))$$

    - "All heroes" in scope of "a cape", i.e., "There is a cape which every hero wears"

$$(\exists y)(Ey \mathbin{\&} (\forall x)(Hx \supset Rxy))$$

$$(\exists y)(\forall x)(Ey \mathbin{\&} (Hx \supset Rxy))$$

- A (probably bad) joke: "Every day, a tourist is mugged on the streets of New York. He's going through a lot of wallets."

11. Multiple quantifiers & The Identity Predicate

j. Donkey sentences

Happy farmers

"Every farmer who owns a donkey is happy"

- Step-by-step symbolization: "All *A*s are *B*s"
- x is a farmer who owns a donkey ...

$$Fx \,\&\, (\exists y)(Dy \,\&\, Oxy)$$

- Every farmer who owns a donkey is happy

$$(\forall x)((Fx \,\&\, (\exists y)(Dy \,\&\, Oxy)) \supset Hx)$$

- Notice how 'a donkey' is bound by an existential here

Unhappy donkeys :(

“Every farmer who owns a donkey beats it”

- Step-by-step symbolization: “All As are Bs”
- x is a farmer who owns a donkey . . .

$$Fx \,\&\, (\exists y)(Dy \,\&\, Oxy)$$

- Every farmer who owns a donkey beats it:

$$(\forall x)((Fx \,\&\, (\exists y)(Dy \,\&\, Oxy)) \supset Bxy)$$

- PROBLEM: ‘y’ is unbound! So this is not a QL sentence. Gasp!

Save the donkeys: a failed attempt

- This was our problem: a donkey lay beaten and unbound:

$$(\forall x)((Fx \,\&\, (\exists y)(Dy \,\&\, Oxy)) \supset Bxy)$$

- Can we simply extend the scope of the existential?

$$(\forall x)(\exists y)((Fx \,\&\, (Dy \,\&\, Oxy)) \supset Bxy)$$

- ‘y’ is now bound, but alas, this sentence is trivially true:
- Provided at least one thing in our UD is not a donkey, that thing makes the antecedent of the conditional false, making the conditional trivially true, for any *x*.
  In particular, our farmer is not a donkey.
  But he still sounds like kind of a jack@$$!

Symbolizing donkey sentences

"Every farmer who owns a donkey beats it"

- When is it false that every farmer who owns a donkey beats it? If there's a farmer who owns a donkey but doesn't beat it. Deny that!

$$\sim(\exists x)(Fx \mathbin{\&} (\exists y)(Dy \mathbin{\&} Oxy \mathbin{\&} \sim Bxy))$$

- For every farmer and every donkey they own: the farmer beats the donkey.

$$(\forall x)(\forall y)((Fx \mathbin{\&} (Dy \mathbin{\&} Oxy)) \supset Bxy)$$

- Every farmer beats every donkey they own.

$$(\forall x)(Fx \supset (\forall y)((Dy \mathbin{\&} Oxy) \supset Bxy))$$

- But what about the case where at least one farmer with a donkey beats only one of his donkeys? #Quitting

12. Metalogic for SL

12. Metalogic for SL

a. A Meta-refresher

SND as a derivation system, provided that...

- As we have seen, Sentential Natural Deduction allows us to derive a conclusion from a set of premises:
  1.) valid argument: conclusion on last line, in scope of just premises
  2.) tautology: on last line in scope of NO premises
  3.) two logically equivalent sentences: (i) their biconditional is a tautology or (ii) derive one from the other and vice versa (which mirrors biconditional introduction!)
- But our derivations are justified only if system SND is *sound*
- And guaranteed to have a derivation for every valid argument only if system SND is *complete*

A tale of three turnstiles: one semantic; two syntactic

- Double Turnstile $\vDash$: logical entailment (indexed to our choice of semantics, i.e. the truth-tables for our connectives)

- Single Turnstile Tree $\vdash_{STD}$: tree-validity in STD (i.e. premises and negated conclusion as root of a tree whose branches all close—recall that this means that $\Gamma \cup \{\sim\Theta\}$ is **tree-inconsistent**)

- Single Turnstile Natural $\vdash_{SND}$: **derivability** in SND

A Tale of Three Turnstiles $\vDash$ the semantic one

- "$\Gamma \vDash \Theta$" means that Γ logically entails Θ
  Whenever the premises in Γ are true, the conclusion Θ is true
- Equivalently: there is no truth-value assignment (TVA) s.t. Γ is satisfied while Θ is false
- Equivalently, this means that $\Gamma \cup \{\sim\Theta\}$ **is unsatisfiable**:
  no TVA satisfies the premises and negated conclusion
- We'll use this last fact A LOT in our proof that SND is complete!

Soundness vs. Completeness

- By proving that our derivation system is *sound*, we show that SND derivations are ‘safe’ (they never lead us astray)
  - **Sound**: If $\Gamma \vdash_{SND} \Theta$, then $\Gamma \vDash \Theta$
  - (syntactic to semantic: i.e. we chose ‘good’ rules!)

- By proving that SND is *complete*, we show truth tables are not needed to demonstrate validity: SND derivations suffice
  - **Complete**: If $\Gamma \vDash \Theta$, then $\Gamma \vdash_{SND} \Theta$
  - (logical entailment is fully covered by our syntactic rules)
  - (Means: we wrote down *enough* rules!)

Some contrasts with our metalogic proofs for trees (STD)

- Recall that to prove the soundness and completeness of our tree system STD, we proved the *contrapositive* of these statements

  - vs. With SND, we'll proceed directly

- With trees, our premise set Γ was finite

  - vs. Here, we'll let Γ be infinite. Although of course, whenever we talk about an SND derivation, this derivation must have a FINITE premise set $\Delta \subseteq \Gamma$ (i.e. a finite list of SL wffs justified by ':PR')

- Is this finiteness restriction a limitation of trees?

- Not in practice: no valid SL argument ever requires infinitely-many premises to entail its conclusion (PS 12 #4)

Semantic entailment for infinitely-many premises

- Let Γ be a possibly infinite set of premises; Θ a conclusion
- Recall: a TVA assigns ‘True’ or ‘False’ to the (infinitely-many) SL atomic wffs
- In the case where Γ is finite, its premises contain finitely-many atomic wffs, so we can restrict a TVA to a row of a truth table
- An argument is **semantically invalid** if there is a TVA that makes each wff in Γ true but which makes Θ false
- In this case we write $\Gamma \nvDash \Theta$
- If there is no such TVA, then $\Gamma \vDash \Theta$

SND derivability for infinitely-many premises

- Θ is **SND-derivable** from Γ provided there is an SND derivation:
  1.) whose starting premises Δ are a finite subset of Γ

  2.) in which Θ appears on its own in the final line

  3.) where Θ is directly next to the main scope line, i.e. only in the scope of the Δ-premises
- In this case, we write $\Gamma \vdash_{SND} \Theta$ (also: $\Delta \vdash_{SND} \Theta$)
- If no such derivation exists, then we say that Θ is NOT SND-derivable from Γ, and we write $\Gamma \nvdash_{SND} \Theta$

12. Metalogic for SL

b. Soundness of System SND

Soundness: Proof Idea and notation

- Subgoal: given any line in an SND derivation, show that the well-formed formula (wff) on that line is entailed by the premises or assumptions accessible from that line
- Let "P_k" be the wff on line k, i.e. the k–th wff in our derivation
- Let "Γ_k" be the set of premises/assumptions accessible on line k, i.e. the set of open assumptions/premises in whose scope P_k lies
- **Subgoal**: given a wff P_k on line k, show that $\Gamma_k \vDash P_k$
- (like with soundness for trees, we reason "from the top down")

Soundness: Proof Strategy

- Recall that SND derivations are defined recursively: from a (possibly empty) set of premises, we have a finite number of rules to add a line

  – These ways include reiteration and an intro and elimination rule for each of our five connectives
- Hence: do induction on the number of lines in an SND derivation
- Show that the base case has the property (line #1)
- Induction hypothesis: assume the property holds for all lines $\leq k$.
- Induction step: show the property holds for line #k+1 (by considering all possible ways line #k+1 could arise)

Let’s get Righteous!

- Say that a line i of a derivation is **righteous** just in case $\Gamma_i \vDash P_i$, i.e. just in case the set of assumptions/premises accessible from i semantically entail the wff on that line.
- Call a derivation *righteous* if every line in it is righteous
- Our goal is to prove that every derivation in SND is righteous!

Do I sound righteous? (from righteousness to soundness)

- Let Γ be any set of SL wffs (possibly infinite)
- If $\Gamma \vdash_{SND} \mathcal{P}$, then by definition there is a derivation whose (finitely-many) premises Δ belong to Γ, such that $\mathcal{P}$ occurs on the final line and lies in the scope of Δ (i.e. $\Delta \vdash_{SND} \mathcal{P}$)
- Then by righteousness, $\Delta \vDash \mathcal{P}$

  - i.e. any TVA that makes Δ true must make $\mathcal{P}$ true
- So there is no truth-value assignment that makes all the sentences in Γ true while making $\mathcal{P}$ false, so $\Gamma \vDash \mathcal{P}$ as well
- So we will have shown **Soundness**: If $\Gamma \vdash_{SND} \mathcal{P}$, then $\Gamma \vDash \mathcal{P}$

Base Case

- **Base case**: for any SND derivation, show that $\Gamma_1 \vDash \mathcal{P}_1$.
- Proof: Γ_1 is the set of premises accessible at line #1, which comprises exactly the wff $\mathcal{P}_1$
- (recall that every premise of a derivation lies in its own scope—

  i.e. these premises be gettin' high off their own supply)
- Clearly, $\mathcal{P}_1 \vDash \mathcal{P}_1$, so $\{\mathcal{P}_1\} \vDash \mathcal{P}_1$
- So line #1 is righteous (i.e. $\Gamma_1 \vDash \mathcal{P}_1$)

Stating the Induction Step

- **Induction Hypothesis**: Assume that every line i for $1 < i \leq k$ is righteous (i.e. that $\Gamma_i \vDash \mathcal{P}_i$)
- Induction step: Consider line #k+1; show that $\Gamma_{k+1} \vDash \mathcal{P}_{k+1}$
- We have 12 cases to consider! 11 of these arise from our 11 SND-sanctioned rules for extending a derivation.
- What is the 12th case?? (We could say 13, but that is BAD LUCK)

Case 1: Premise or Assumption

- **Case 1**: $\mathcal{P}_{k+1}$ is a premise ($:PR$) or a subproof assumption ($:AS$). Show that $\Gamma_{k+1} \vDash \mathcal{P}_{k+1}$
- Either way, $\mathcal{P}_{k+1} \in \Gamma_{k+1}$ (since every premise and assumption lies within its own scope)
- So given a TVA that makes every sentence in Γ_{k+1} true, this TVA must make $\mathcal{P}_{k+1}$ true
- So $\Gamma_{k+1} \vDash \mathcal{P}_{k+1}$; so this case be righteous!

Case 2: Reiteration

- **Case 2**: $\mathcal{P}_{k+1}$ arises from an application of rule R, reiteration
- Then wff $\mathcal{P}_{k+1}$ appears on an earlier line #i as the wff $\mathcal{P}_i$
- By the induction hypothesis, line #i is righteous, so $\Gamma_i \vDash \mathcal{P}_i$.

  -Hence, we also have $\Gamma_i \vDash \mathcal{P}_{k+1}$ (since $\mathcal{P}_i = \mathcal{P}_{k+1}$)
- To apply rule R, $\mathcal{P}_{k+1}$ must lie to the right of line #i's rightmost scope line $\Rightarrow \Gamma_i \subseteq \Gamma_{k+1}$ (i.e., all of the premises/assumptions accessible at line #i must also be accessible at line #k+1).
- Since $\Gamma_i \vDash \mathcal{P}_{k+1}$ and $\Gamma_i \subseteq \Gamma_{k+1}$, we have $\Gamma_{k+1} \vDash \mathcal{P}_{k+1}$
- Draw a schematic derivation to better understand $\Gamma_i \subseteq \Gamma_{k+1}$!

Case 3: Conjunction Introduction (Things be heating up—finally!)

- **Case 3**: $\mathcal{P}_{k+1} := (\mathcal{Q} \,\&\, \mathcal{R})$ arises from an application of rule $\&$I
- Then on two earlier lines #h and #j, $\mathcal{Q}$ and $\mathcal{R}$ appear, respectively
- By the IH, both of these lines are righteous, so $\Gamma_h \vDash \mathcal{Q}$ and $\Gamma_j \vDash \mathcal{R}$
- By rule $\&$I, both these lines must be accessible on line #k+1
- So $\Gamma_h \cup \Gamma_j \subseteq \Gamma_{k+1}$ (i.e. both Γ_h and Γ_j are subsets of Γ_{k+1})
- Hence, any TVA that satisfies Γ_{k+1} must satisfy both Γ_h and Γ_j, and hence satisfy $\mathcal{Q}$ and also satisfy $\mathcal{R}$
- Thus, any TVA that satisfies Γ_{k+1} satisfies $(\mathcal{Q} \,\&\, \mathcal{R})$
- So $\Gamma_{k+1} \vDash \mathcal{P}_{k+1}$

Case 4: Conjunction Elimination

- **Case 4**: $\mathcal{P}_{k+1}$ arises from an application of rule &E

  I'm about to eliminate this proof, son!
- Then there is an earlier line #h of the form $\mathcal{P}_{k+1} \& \mathcal{Q}$ or $\mathcal{Q} \& \mathcal{P}_{k+1}$
- By the IH, line #h is righteous, so $\Gamma_h \vDash \mathcal{P}_h$
- Since line #h is accessible at line #k+1, $\Gamma_h \subseteq \Gamma_{k+1}$
- So any TVA that satisfies Γ_{k+1} also satisfies Γ_h and thereby makes true $\mathcal{P}_h$
- By the truth conditions for conjunctions, any TVA that satisfies $\mathcal{P}_h$ satisfies both conjuncts, in particular $\mathcal{P}_{k+1}$
- So $\Gamma_{k+1} \vDash \mathcal{P}_{k+1}$ and line #k+1 is righteous

Case 8: Conditional Introduction

- **Case 8**: $\mathcal{P}_{k+1}$ arises from rule $\supset$I, which involves a subproof!
- $\mathcal{P}_{k+1}$ must be of the form $\mathcal{Q} \supset \mathcal{R}$ (**draw derivation** to define terms)
- NTS: $\Gamma_{k+1} \vDash \mathcal{Q} \supset \mathcal{R}$ given that $\Gamma_h \vDash \mathcal{Q}$ and $\Gamma_j \vDash \mathcal{R}$, by Ind. Hyp.
- Proceed by cases: either Γ_{k+1} satisfies $\mathcal{Q}$ or it doesn't:
- If Γ_{k+1} does not satisfy $\mathcal{Q}$, then it trivially satisfies $\mathcal{Q} \supset \mathcal{R}$
- Otherwise, Γ_{k+1} satisfies $\mathcal{Q}$. Since $\Gamma_j \subseteq \Gamma_{k+1} \cup \{\mathcal{Q}\}$, this means that Γ_j is satisfied in this case. Then since line #j is righteous, we have $\Gamma_{k+1} \cup \{\mathcal{Q}\} \vDash \mathcal{R}$. So in this case, Γ_{k+1} satisfies $\mathcal{Q} \supset \mathcal{R}$ as well.
- So in either case, $\Gamma_{k+1} \vDash \mathcal{P}_{k+1}$

Case 9: Negation Introduction

- **Case 8**: $\mathcal{P}_{k+1}$ arises from rule $\sim$I, using a subproof!
- $\mathcal{P}_{k+1}$ must be of form $\sim\mathcal{Q}$; **draw derivation to define lines**
- NTS: $\Gamma_{k+1} \vDash \sim\mathcal{Q}$ given that $\Gamma_h \vDash \mathcal{Q}$, $\Gamma_j \vDash \mathcal{R}$ **and** $\Gamma_m \vDash \sim\mathcal{R}$ (by IH)
- Notice that Γ_j and Γ_m are both subsets of $\Gamma_{k+1} \cup \{\mathcal{Q}\}$

  Hence, $\Gamma_{k+1} \cup \{\mathcal{Q}\}$ entails both $\mathcal{R}$ and $\sim\mathcal{R}$ as well.

  Thus, any TVA that satisfies $\Gamma_{k+1} \cup \{\mathcal{Q}\}$ must make both $\mathcal{R}$ and $\sim\mathcal{R}$ true, which is impossible (i.e. there can be no such TVA).

  $\Rightarrow \Gamma_{k+1} \cup \{\mathcal{Q}\}$ is unsatisfiable. Hence, $\Gamma_{k+1} \vDash \sim\mathcal{Q}$

12. Metalogic for SL

c. Completeness of System SND

Semantic vs. Syntactic Consistency

- We will appeal to two distinct notions of consistency throughout
- One is **semantic**: this is the notion we are already familiar with: there is a TVA that **satisfies** every sentence in the set
- We introduce a new **syntactic** notion of consistency relative to our SND derivation system:

  - a set of SL wffs is **SND-consistent** provided that you can't derive contradictory sentences from it in SND
- Core proof idea: we'll show that if a set of sentences is **consistent-in-SND**, then it is also semantically consistent (i.e. **satisfiable**). So by the contrapositive: if a set is **un**satisfiable, then it is **in**consistent-in-SND.

Semantic: Satisfiable (truth-functionally consistent)

- Recall: a set of SL sentences is **satisfiable** provided there is a TVA that makes all of them true
- This is a *semantic* notion of consistency
- i.e. **truth-functionally consistent**
- Contrast this with the syntactic notion of **consistency in SND**:

Syntactic: (In)consistent-in-SND (derivationally consistent)

- Let Γ be a (possibly infinite) set of SL wffs
- **Inconsistent-in-SND**: from premises in Γ, we can derive contradictory formulas R and $\sim R$ in the scope of the main scope line (i.e. these premises)
- **Consistent-in-SND**: Γ is not SND-inconsistent, i.e. there is no derivation from premises in Γ resulting in contradictory formulas within the main scope
- Other words we might use for these concepts: SND-inconsistent, derivationally-inconsistent, SND-consistent, etc.
- Just remember: this syntactic notion has nothing to do with truth value assignments!

Proof Sketch

- Goal: prove the completeness of SL: for every SL wff $\mathcal{P}$ and every set Γ of SL sentences, if $\Gamma \vDash \mathcal{P}$ then $\Gamma \vdash_{SND} \mathcal{P}$
- So assume that $\Gamma \vDash \mathcal{P}$.
- Recall from week 5: this means that $\Gamma \cup \{\sim\mathcal{P}\}$ is semantically inconsistent (i.e. **unsatisfiable**):
  no TVA satisfies the premises and negated conclusion
- We now appeal to a **Consistency lemma** that is the heart of the enterprise: any SND-consistent set of SL sentences is satisfiable (i.e. semantically consistent)

Proof Sketch: Using the consistency lemma

- **Consistency lemma**: any SND-consistent set of SL sentences is satisfiable
- **Contrapositive** of CL: any set of SL sentences that is Unsatisfiable is SND-Inconsistent
- From $\Gamma \vDash \mathcal{P}$ we know that $\Gamma \cup \{\sim\mathcal{P}\}$ is unsatisfiable
- So by the contrapositive of CL, we see that $\Gamma \cup \{\sim\mathcal{P}\}$ is SND-inconsistent
- This means that we can derive a pair of contradictory sentences R and $\sim R$ from $\Gamma \cup \{\sim\mathcal{P}\}$! So using the power of negation elimination, we can derive $\mathcal{P}$ from Γ, i.e. $\Gamma \vdash \mathcal{P}$. So we are 'done'!

Negation Elimination Refresher (book's claim 6.4.4)

- Claim: if $\Gamma \cup \{\sim\mathcal{P}\}$ is **SND-inconsistent**, then $\Gamma \vdash \mathcal{P}$
- Proof: starting with (finitely-many) premises Δ from Γ, introduce $\sim\mathcal{P}$ as a subproof assumption for negation elimination
- Since $\Gamma \cup \{\sim\mathcal{P}\}$ is SND-inconsistent, we can derive a contradictory pair R and $\sim R$ within the scope of wffs in $\Delta \cup \{\sim\mathcal{P}\}$
- Then discharge this assumption $\sim\mathcal{P}$ by negation elimination, writing $\mathcal{P}$, now in the scope of Δ. So $\Delta \vdash \mathcal{P}$
- Since $\Delta \subseteq \Gamma$, we have $\Gamma \vdash \mathcal{P}$

Core subgoal: Prove consistency lemma (book's 6.4.2)

- So all we have to do is prove the **consistency lemma**: any SND-consistent set of SL sentences is satisfiable
- We'll prove this lemma in three 'stages':
- The first two are straightforward: given an SND-consistent set Γ, we construct a **superset** Γ^* that is *maximally SND-consistent*
- In the third stage, we show that any maximally SND-consistent set is satisfiable: we use maximal consistency to construct a TVA that satisfies every sentence in Γ^*
- Since by construction $\Gamma \subseteq \Gamma^*$, this TVA satisfies Γ as well.
- (The idea in the third stage is similar to what we did with trees: use a syntactic consistency property to construct a TVA that satisfies a set of wffs: with trees we had 'complete open branches'; here we have maximal–SND–consistency)
- The third stage comprises a tedious lemma and induction! PS12 problems 2 and 3 provide practice with this tedium!

Maximally SND-consistent

- A set Γ^* of SL wffs is **maximally SND-consistent** provided that:
  1.) Γ^* is SND-consistent (i.e. can't derive contradictory sentences)
  2.) adding **any** additional wff to Γ^* would result in an SND-inconsistent set
- i.e. for any $P \notin \Gamma^*$, $\{P\} \cup \Gamma^*$ is SND-inconsistent
- Motivation: it is straightforward (but tedious) to show that a maximally SND-consistent set is semantically consistent

  - Moreover, every SND-consistent set is a subset of a maximally SND-consistent set.

  - So we piggyback on an appropriate Γ^* to show that any SND-consistent set Γ is also satisfiable

Stage 1: Constructing Γ^*

- Let Γ be an SND-consistent set of SL wffs (possibly infinite)
- To construct Γ^*, we first **enumerate** the SL wffs, so that every SL wff is associated with a unique positive integer $\{1, 2, 3, \ldots\}$
- Then consider the first wff 'A' in our enumeration.
  If A can be added to Γ without the resulting set being SND-inconsistent, then let $\Gamma_1 := \Gamma \cup \{A\}$.
- Otherwise, let $\Gamma_1 := \Gamma$ (so that Γ_1 stays SND-consistent)
- Then, proceed to the second wff in our enumeration.
  If it can be added to Γ_1 without the new set being SND-inconsistent, let Γ_2 be the result. Otherwise, let $\Gamma_2 := \Gamma_1$
- Γ^* is the result of 'doing' this procedure for every SL wff
- More precisely, $\Gamma^* := \bigcup_{k=1}^{\infty} \Gamma_k$

Enumeration (lexical ordering)

- Analogy: we can enumerate words by length, using their alphabetical order to break ties
- Can do the same for SL wffs by stipulating an ‘alphabetical order’:
- $\sim, \vee, \&, \supset, \equiv, (,), 0, 1, \ldots, 9, A, B, \ldots, Z$
- Each symbol is assigned an **index** between ‘10’ and ‘55’
- Then each SL wff corresponds to a unique positive integer, constructed by replacing each symbol in the wff with its index, from left to right.
- So with our ordering, ‘A’ is the first wff; ‘B’ the second ... up to Z, and then we hit $\sim A$ ($\mapsto$ 1030), then $\sim B$ ($\mapsto$ 1031), etc.

Stage 2: Γ^* is maximally SND-consistent

- This requires proving two claims (from definition of M-SND-C):

  1.) Γ^* is consistent in SND

  2.) Adding any additional wff to Γ^* would result in an **SND-inconsistent** set

- We prove these in turn

Stage 2 (i): Γ^* is SND-consistent

- Assume for *reductio* that Γ^* is inconsistent in SND
- Then there would be an SND derivation with finite premise set $\Delta \subset \Gamma^*$ that derives a contradictory pair R and $\sim R$
- Since Δ is finite, there exists some $k \in \mathbb{N}$ s.t. $\Delta \subset \Gamma_k$.
  So then this Γ_k would be SND-inconsistent.
- Yet, we constructed each Γ_k such that each is SND-consistent:
  - In general, if P_k is the k-th sentence in our enumeration, then Γ_{k+1} is $\Gamma_k \cup \{P_k\}$ provided that $\Gamma_k \cup \{P_k\}$ is SND-consistent; otherwise, Γ_{k+1} equals Γ_k (so SND-consistent either way)
- Hence, Γ^* must be SND-consistent, on pain of *reductio*
- (note: the book's proof, p. 256, is way more complicated than necessary...)

Stage 2 (ii): Γ^* is maximally SND-consistent

- Assume for *reductio* that Γ^* weren't maximally SND-consistent, despite being SND-consistent
- i.e. assume *it is not the case that* for all additional wff, adding it to Γ^* would result in an SND-inconsistent set
  $\Rightarrow$ there exists a wff $\mathcal{Q}$ that we could add to Γ^* while preserving SND-consistency (i.e. there would be some wff that we neglected that could make Γ^* an even 'bigger' SND-consistent set)
- Yet, $\mathcal{Q}$ would appear in our enumeration as some wff P_k, 'considered' at the k-th stage of our construction of Γ^*.
- So if $\mathcal{Q}$ isn't in Γ^*, then this is because adding it 'would have' made $\Gamma_k \subset \Gamma^*$ SND-inconsistent.
  So $\{\mathcal{Q}\} \cup \Gamma^*$ must be SND-inconsistent (*reductio*!)
- So we can't add any $\mathcal{Q}$ to Γ^* while preserving SND-consistency

Stage 3: The Maximal Consistency Lemma (book's 6.4.8)

- **Maximal Consistency Lemma**: any set that is maximally–SND–consistent is satisfiable
- So there exists a TVA that satisfies every sentence in Γ^*. We construct this TVA, calling it "$\mathcal{I}$" (the book calls it $\mathbf{A}^*$)
- Proof idea: since Γ^* is M–SND–C, for any wff $\mathcal{Q}$, either $\mathcal{Q} \in \Gamma^*$ or $\sim\mathcal{Q} \in \Gamma^*$ (you're either in the club or your 'nemesis' is!)

  This holds in particular for each atomic wff
- Define the TVA $\mathcal{I}$ such that $\mathcal{I}(B) = \mathit{True}$ iff atomic $B \in \Gamma^*$
- Then by the recursive structure of SL wffs, $\mathcal{I}(\mathcal{Q}) = \mathit{True}$ iff $\mathcal{Q} \in \Gamma^*$

Stage 3 (i): the Membership Lemma (book's 6.4.11)

- To induct on SL, we first show some constraints on Γ^* membership
- Basically, Γ^* is like a club with a bouncer who enforces maximal consistency. Before the bouncer lets a wff into Γ^*, he checks who else is in the club
- **Membership Lemma** for club Γ^*: if $\mathcal{P}$ and $\mathcal{Q}$ are SL wffs, then:
  - a.) $\sim\mathcal{P} \in \Gamma^*$ if and only if $\mathcal{P} \notin \Gamma^*$
  - b.) $\mathcal{P} \,\&\, \mathcal{Q} \in \Gamma^*$ if and only if both $\mathcal{P} \in \Gamma^*$ and $\mathcal{Q} \in \Gamma^*$
  - c.) $\mathcal{P} \vee \mathcal{Q} \in \Gamma^*$ if and only if either $\mathcal{P} \in \Gamma^*$ or $\mathcal{Q} \in \Gamma^*$
  - d.) $\mathcal{P} \supset \mathcal{Q} \in \Gamma^*$ if and only if either $\mathcal{P} \notin \Gamma^*$ or $\mathcal{Q} \in \Gamma^*$
  - e.) $\mathcal{P} \equiv \mathcal{Q} \in \Gamma^*$ iff either (i) $\mathcal{P} \in \Gamma^*$ and $\mathcal{Q} \in \Gamma^*$ or (ii) $\mathcal{P} \notin \Gamma^*$ and $\mathcal{Q} \notin \Gamma^*$
- Notice how these syntactic constraints mirror truth–conditions!
- Moral: We all want to belong, but sometimes our enemies get in the way!

Stage 3 (i): Key Fact aka The Door lemma (book's 6.4.9)

- To prove the membership lemma's cases (a)-(e), we'll use another lemma (hint: it's lemmas all the way down):
- **The Door**: if $\Gamma \vdash P$, and Γ^* is a maximally SND–consistent superset of Γ, then $P \in \Gamma^*$
  (mnemonic: "$\Gamma \vdash P$" pushes P through the door!)
- Proof: first, assume that $\Gamma \vdash P$ (we'll use this fact below)
- Next, assume for *reductio* that $P \notin \Gamma^*$. Then since Γ^* is maximally SND–consistent, $\Gamma^* \cup \{P\}$ must be inconsistent in SND.
- Hence, by negation introduction, $\Gamma^* \vdash \sim P$
- By assumption, $\Gamma \vdash P$, so also $\Gamma^* \vdash P$, since $\Gamma \subseteq \Gamma^*$
- So Γ^* derives both P and $\sim P$. *Reductio*! (since Γ^* is M–SND–C)
- Hence, if $\Gamma \vdash P$ and $\Gamma \subseteq \Gamma^*$, then P must belong to Γ^*

Membership Lemma: Case (a)

- **Case (a)**: $\sim\mathcal{P} \in \Gamma^*$ if and only if $\mathcal{P} \notin \Gamma^*$
- Two directions to prove:

  $\Rightarrow$: Assume $\sim\mathcal{P} \in \Gamma^*$. Then if $\mathcal{P}$ were in Γ^*, we could derive contradictory sentences.

  So since Γ^* is SND-consistent, we must have $\mathcal{P} \notin \Gamma^*$

  $\Leftarrow$: Assume $\mathcal{P} \notin \Gamma^*$. Then adding $\mathcal{P}$ to Γ^* results in an SND-inconsistent set. Hence, there is some finite subset $\Delta \subset \Gamma^*$ s.t. $\Delta \cup \{\mathcal{P}\}$ is SND-inconsistent (i.e. derives contradictory sentence pair).
- So by negation introduction, $\Delta \vdash \sim\mathcal{P}$
- So by The Door lemma, $\sim\mathcal{P} \in \Gamma^*$

Membership Lemma: Cases (b)–(e)

- See the book for cases (b) ($\mathcal{P} \,\&\, \mathcal{Q}$) and (d) ($\mathcal{P} \supset \mathcal{Q}$)
- Case (c) is PS12 #2: $\mathcal{P} \vee \mathcal{Q} \in \Gamma^*$ if and only if either $\mathcal{P} \in \Gamma^*$ or $\mathcal{Q} \in \Gamma^*$
- We skip case (e) ($\mathcal{P} \equiv \mathcal{Q}$) because . . . **YOLO**

Stage 3 (ii): Induction on SL (i.e. we be clubbin')

- Goal: construct a TVA $\mathcal{I}$ that satisfies the M-SND-C set Γ^*

  Suffices to construct $\mathcal{I}$ s.t. $\mathcal{I}(\mathcal{Q}) = \mathit{True}$ iff $\mathcal{Q} \in \Gamma^*$, $\forall \mathcal{Q} \in$ SL.
  Say that a wff is "**clubbin'** " whenever it meets this property
- Define $\mathcal{I}$ such that $\mathcal{I}(B) = \mathit{True}$ iff atomic $B \in \Gamma^*$
- **Base case**: each atomic wff is true on $\mathcal{I}$ iff it belongs to Γ^* (i.e. the atomics be clubbin')
- (Strong) **Induction hypothesis**: assume every SL wff with 1 to k–many connectives is clubbin'
- Induction step: show that an arbitrary SL wff with k+1–many connectives is clubbin'

Base Case

- Need to show **TWO** directions!:
- **Base case**: each atomic wff is true on $\mathcal{I}$ **iff** it belongs to Γ^*
- Recall that we defined $\mathcal{I}$ such that $\mathcal{I}(B) = \mathit{True}$ **iff** atomic $B \in \Gamma^*$
- So both directions are met by construction
- We proceed to do induction using our SL induction schema: an arbitrary sentence $\mathcal{P}$ with k+1–many connectives has one of five forms, coming from our five connectives.

Induction on SL: Case 1

- **Case 1**: $\mathcal{P}$ has the form $\sim\mathcal{Q}$, where since $\mathcal{Q}$ has k-connectives, it is clubbin by the IH (i.e. $\mathcal{I}(\mathcal{Q}) = 1$ if and only if $\mathcal{Q} \in \Gamma^*$)
- NTS: (i) (the $\Rightarrow$direction) if $\mathcal{I}(\mathcal{P}) =$ *True* then $\mathcal{P} \in \Gamma^*$ and (ii) (the $\Leftarrow$direction) if $\mathcal{P} \in \Gamma^*$, then $\mathcal{I}(\mathcal{P}) =$ *True*

  (*Alternative (ii)*: show contrapositive: if $\mathcal{I}(\mathcal{P}) = 0$, then $\mathcal{P} \notin \Gamma^*$)

$\Rightarrow$ if $\mathcal{I}(\mathcal{P}) = 1$, then $\mathcal{I}(\mathcal{Q}) = 0$. Since $\mathcal{Q}$ is clubbin', we have $\mathcal{Q} \notin \Gamma^*$.

By Membership lemma (a), $\sim\mathcal{Q} \in \Gamma^*$, so $\mathcal{P} \in \Gamma^*$

$\Leftarrow$ if $\mathcal{P} \in \Gamma^*$, then $\sim\mathcal{Q} \in \Gamma^*$. So by Membership lemma (a), $\mathcal{Q} \notin \Gamma^*$.

Since $\mathcal{Q}$ is clubbin', we have $\mathcal{I}(\mathcal{Q}) = 0$.

So by the truth conditions for negation, $\mathcal{I}(\mathcal{P}) = 1$

Induction on SL: Cases 2–5

- Need to show: $\mathcal{P}$ be clubbin', i.e. $\mathcal{I}(\mathcal{P}) = \mathit{True}$ iff $\mathcal{P} \in \Gamma^*$, where $\mathcal{P}$ is arbitrary SL wff with k+1–many connectives
- Induction hypothesis: assume every SL wff with 1 to k–many connectives is clubbin'
- **Case 2**: $\mathcal{P}$ has the form $\mathcal{Q} \,\&\, \mathcal{R}$
- **Case 3** is PS12 #3: $\mathcal{P}$ has the form $\mathcal{Q} \vee \mathcal{R}$
- Case 4: $\mathcal{P}$ has the form $\mathcal{Q} \supset \mathcal{R}$ (see book p.260!)
- Case 5: $\mathcal{P}$ has the form $\mathcal{Q} \equiv \mathcal{R}$ (we'll do this case if and only if we accomplish all other goals in our lives)

Reminder for Josh!

- If we actually make it this far, give hints on PS12 completeness question ($P \vee Q$)! or do Case (d), which is most analogous
- If the people don't want these hints, then clearly they're already complete!
- "The customer is always right!"
- (Schematize this sentence in quantifier logic)

13. Metalogic for QL

Soundness vs. Completeness

- Let Γ be any set of *sentences* of QL and Θ any sentence of QL.
- By proving that our derivation system is *sound*, we show that QND derivations are 'safe' (they preserve truth)
  - **Sound**: If $\Gamma \vdash_{QND} \Theta$, then $\Gamma \vDash \Theta$
  - (syntactic to semantic: i.e. we chose 'good' rules!)
- By proving that QND is *complete*, we show that reasoning about arbitrary models is not needed to demonstrate validity: QND derivations suffice
  - **Complete**: If $\Gamma \vDash \Theta$, then $\Gamma \vdash_{QND} \Theta$
  - (logical entailment is fully covered by our syntactic rules)
  - (Means: we wrote down *enough* rules!)

13. Metalogic for QL

a. Truth and Satisfaction in QL

Recap: models and interpretations

- Let $\mathfrak{L}$ be a first-order language, containing constants and k-place predicates (e.g. the language of QL)
  - recall that the atomic sentences of SL are 0th-place predicates
- An $\mathfrak{L}$-model $\mathfrak{M} := (D, I)$ consists of
  1. A non-empty set D of objects, called the domain of $\mathfrak{M}$
  2. A map I (the *interpretation* of $\mathfrak{M}$), which maps the vocabulary of $\mathfrak{L}$ to objects and ordered pairs from D as follows:
     - For each constant $c \in \mathfrak{L}$, $I(c)$ is an element of D, called the *referent* or denotation of c
     - For each k-place predicate P of $\mathfrak{L}$, $I(P)$ is a set of ordered k-tuples of objects in D, called the *extension* of P
- Our text uses ‘models’ and ‘interpretations’ interchangeably, but the above disambiguation is convenient

Truth in a Model: simple examples

- Consider a simple language $\mathfrak{L}$ comprising a one-place predicate P, a two-place predicate R, and constants a and b.
- Fix an $\mathfrak{L}$-model $\mathfrak{M} := (D, I)$, e.g. our D could be $\mathbb{N}$.
  I is what we would punch into *Carnap* on PS9
- Pa is true in $\mathfrak{M}$ provided that the object $I(a)$ has the property $I(P)$, i.e. $I(a)$ lies in the extension of P. Then we'll write $\mathfrak{M} \vDash Pa$
- Rab is true in $\mathfrak{M}$ provided that the objects $I(a)$ and $I(b)$ stand in relation R. In this case, we'll write $\mathfrak{M} \vDash Rab$

Satisfaction in a model: simple example

- What to say about something with free variables, such as Px?
- This is a wff of QL but not a sentence (it is neither true nor false in a model)
- Idea: for each object r in D, we know whether Px *would be true* if we replaced r for x, i.e. if x stood for r
  (b/c we know the extension of P, i.e. all the objects that are P)
- *Shorthand*:if Pc is true in $\mathfrak{M}$, then $I(c)$ satisfies Px in $\mathfrak{M}$
- *Longhand*: define a variable assignment $\mathbf{d}_I$ that maps variables to objects. Then $\mathbf{d}_I$ **satisfies** Px provided that $\mathbf{d}_I(x)$ has property I(P). We can write $\mathfrak{M}_{\mathbf{d}_I} \vDash Px$

Satisfaction for Atomic Wffs

- **Atomic wffs**: Suppose $\mathcal{Q}$ is atomic. Then $\mathcal{Q}$ is of the form $\mathcal{P}t_1 \ldots t_k$ where each t_i is a term, i.e. a constant or a variable.

- I assigns constants to objects t_i^D; d_I maps variables to objects t_i^D (the text calls these denotations under I or d_I "$\mathrm{den}_{I,d_I}(t_i)$")

- d_I satisfies $\mathcal{Q}$ provided that the k–tuple of these objects $\langle t_1^D, \ldots, t_k^D \rangle$ lies in the extension of $\mathcal{Q}$, i.e. in $I(\mathcal{Q})$

Satisfaction for Quantified Wffs

- For $r \in D$ we write $d_I[r/x]$ for the assignment that agrees with d_I except necessarily assigning r for x
- **Existentially Quantified**: Suppose we have a wff of the form $(\exists x)\mathcal{Q}$. Then d_I satisfies $(\exists x)\mathcal{Q}$ provided there is SOME object $r \in D$ such that $d_I[r/x]$ satisfies $\mathcal{Q}$
  - Intuition: provided there's at least one thing you can plug in for x such that $\mathcal{Q}$ comes out true
- **Universally Quantified**: Suppose we have a wff of the form $(\forall x)\mathcal{Q}$. Then d_I satisfies $(\forall x)\mathcal{Q}$ provided $d_I[r/x]$ satisfies $\mathcal{Q}$ for EACH object $r \in D$
  - Intuition: no matter what you plug in for x, $\mathcal{Q}$ comes out true

From Satisfaction to Truth

- Focus on the sentences of QL, which have no free variables
- Lemma: given a model $\mathfrak{M} = (D, I)$ and a QL *sentence* $\mathcal{P}$, either all variable assignments d_I satisfy $\mathcal{P}$ or none do.
- Hence, we can define truth in a QL–model as follows:
- A sentence $\mathcal{P}$ of QL is **true** on model $\mathfrak{M}$ iff some variable assignment d_I satisfies $\mathcal{P}$ in $\mathfrak{M}$
- A sentence $\mathcal{P}$ of QL is **false** on model $\mathfrak{M}$ otherwise, i.e. if no variable assignment d_I satisfies $\mathcal{P}$ in $\mathfrak{M}$

Shorthand: Truth of quantified sentences

- $(\exists x)\,\mathcal{A}x$ is true iff $\mathcal{A}x$ is **satisfied** by **at least one** object in D
  - $r = I(c)$ satisfies $\mathcal{A}x$ in $\mathfrak{M}$ iff $\mathcal{A}c$ is true in $\mathfrak{M}$
  - e.g. there is at least one object in the domain that is an $\mathcal{A}$
  - Formally, there is a variable assignment d_I with at least one variant $d_I[r/x]$ s.t. $d_I[r/x]$ satisfies $\mathcal{A}x$
- $(\forall x)\,\mathcal{A}x$ is true iff $\mathcal{A}x$ is **satisfied** by **every** object in the domain
  - e.g. everything in D is an $\mathcal{A}$
  - Formally, there is a variable assignment d_I such that for each $r \in D$, each variant $d_I[r/x]$ satisfies $\mathcal{A}x$

Examples of Shorthand

- $(\exists x)(\mathcal{A}x \,\&\, \mathcal{B}x)$ is true iff some object satisfies '$\mathcal{A}x \,\&\, \mathcal{B}x$'
  - o satisfies '$\mathcal{A}x \,\&\, \mathcal{B}x$' iff it satisfies both $\mathcal{A}x$ and $\mathcal{B}x$
- $(\forall x)(\mathcal{A}x \supset \mathcal{B}x)$ is true iff **every** object satisfies '$\mathcal{A}x \supset \mathcal{B}x$'
  - o satisfies '$\mathcal{A}x \supset \mathcal{B}x$' iff either
    - o does not satisfy $\mathcal{A}x$ (vacuously true conditional)

      or
    - o does satisfy $\mathcal{B}x$

Semantic Notions in QL

- Given a premise-set Γ of QL-*sentences* and a conclusion *sentence* $\mathcal{Q}$, we have the following semantic notions:

- **Entailment**: Γ QL-entails $\mathcal{Q}$ provided that there is no QL-model $\mathfrak{M}$ where Γ is true but $\mathcal{Q}$ is false. We write $\Gamma \vDash \mathcal{Q}$
  - we say that the argument from Γ to $\mathcal{Q}$ is **QL-valid**

- **Satisfiability**: we say that a set of sentences Γ is jointly **satisfiable** (aka QL-consistent) provided that there exists at least one QL-model $\mathfrak{M}$ where each sentence in Γ is true

13. Metalogic for QL

b. Recap: Substitution Instances

(Full) Substitution Instances

- "$\mathcal{Q}[c/x]$" is the sentence you get from $(\forall x)\mathcal{Q}$ or $(\exists x)\mathcal{Q}$ by dropping the quantifier and putting c in place of **every** x in $\mathcal{Q}$

- The other variables are untouched!

- Read "$[c/x]$" as saying "substitute c for every x", i.e. all the x's are replaced by c's!

Some Examples of Substitution Instances

- Instances of $(\forall y)$Hy:
  - Ha, Hb, Hm_{11}
- Instances of $(\exists z)$Haz:
  - Haa, Hab, Haj_3
- Instances of $(\exists z)(Hz \,\&\, Fzz)$:
  - Remember to replace **EVERY** occurance of z with the chosen constant:
  - $(Ha \,\&\, Faa)$, $(Hc \,\&\, Fcc)$
  - The following are **NOT** substitution instances:
  - $(Ha \,\&\, Faz)$, $(Hy \,\&\, Faa)$, $(Ha \,\&\, Fab)$

Partial Substitution Instances

- For Existential Introduction, we can use a partial substitution instance of the wff $\mathcal{Q}$:
- "$\mathcal{Q}\lceil \chi / c \rceil$" indicates that the variable χ replaces some but not necessarily all occurrences of the constant c in $\mathcal{Q}$.
- You can decide which occurrences of c to replace and which to leave in place

Examples of Partial Substitution Instances!

- '$\mathcal{Q}[\chi/c]$' indicates that the variable χ does not need to replace all occurrences of the constant c in $\mathcal{Q}$

1	Rdd	
2	$(\exists x)Rxx$	:1 ∃I
3	$(\exists x)Rxd$	:1 ∃I
4	$(\exists z)Rdz$	:1 ∃I
5	$(\exists y)(\exists z)Ryz$	:4 ∃I

Existential Introduction (∃I)

m	$\mathcal{Q}$	
⋮	⋮	
s	$(\exists \chi)\mathcal{Q}[\chi/c]$	:m ∃I

– Note: since $\mathcal{Q}$ is a sentence, and by our recursion clause for wff, χ cannot occur in $\mathcal{Q}$.

Substitution Lemma (Logic Book 11.1.1)

- Consider $\mathcal{Q} := Fxx$. Then $\mathcal{Q}[c/x] = Fcc$
- "variant $d_I[I(c)/x] = d_I[r/x]$ satisfies Fxx" means that when we assign x to $r = I(c)$, Fcc is true ($\langle r, r\rangle \in Extension(F)$)
- "d_I satisfies $\mathcal{Q}[c/x]$" means roughly that whatever objects d_I assigns variables, the result lies in the Extension of $\mathcal{Q}$
- Note that since x doesn't appear in Fcc, d_I treats Fcc just like $d_I[I(c)/x]$ treats Fxx
- "d_I satisfies $\mathcal{Q}[c/x]$" is equivalent to "$d_I[I(c)/x]$ satisfies $\mathcal{Q}$"
- **Substitution Lemma**: let $\mathcal{Q}$ be a wff of QL. The variable assignment d_I satisfies $\mathcal{Q}[c/\chi]$ if and only if $d_I[I(c)/\chi]$ satisfies $\mathcal{Q}$

13. Metalogic for QL

c. QL rules recap

Rules for the Universal Quantifier

Universal Elimination (∀E)

$$\begin{array}{l|ll} m & (\forall \mathcal{x})\mathcal{Q} & \\ \vdots & \quad\vdots & \\ s & \mathcal{Q}[c/\mathcal{x}] & :m\ \forall E \end{array}$$

- Note that you replace **EVERY** instance of $\mathcal{x}$ with c

- Notation: $\mathcal{Q}[c/\mathcal{x}]$

- read "c for $\mathcal{x}$"

Universal Introduction (∀I)

$$\begin{array}{l|ll} m & \mathcal{Q} & \\ \vdots & \quad\vdots & \\ s & (\forall \mathcal{x})\mathcal{Q}[\mathcal{x}/c] & :m\ \forall I \end{array}$$

Provided that both
(i) c does not occur in any other undischarged assumptions that $\mathcal{Q}$ is in the scope of.
(ii) $\mathcal{x}$ does not occur already in $\mathcal{Q}$.

Rules for the Existential Quantifier

Existential Introduction ($\exists$I)

$$\begin{array}{l|ll}
m & \mathcal{Q} & \\
\vdots & \quad\vdots & \\
s & (\exists \chi)\mathcal{Q}\lceil \chi / c \rceil & :m\ \exists\text{I}
\end{array}$$

- **Provided that** χ does not occur already in $\mathcal{Q}$.

- As indicated by $\lceil \chi / c \rceil$, χ **may** replace **just some** occurrences of c

Existential Elimination ($\exists$E)

$$\begin{array}{l|l|ll}
m & \multicolumn{2}{l}{(\exists \chi)\mathcal{Q}} & \\
 & \multicolumn{2}{l}{\quad\vdots} & \\
n & & \underline{\mathcal{Q}[c/\chi]} & :\text{AS for } \exists\text{E} \\
 & & \quad\vdots & \\
s & & \Psi & \\
s+1 & \multicolumn{2}{l}{\Psi} & :m,\ n\text{–}s\ \exists\text{E}
\end{array}$$

Simplified: **provided that** c doesn't occur **anywhere else outside** the subproof

Motivating these restrictions on various rules!

- We'll now see why the rules $\forall I$ and $\exists E$ require us to follow the stated, non-trivial restrictions
- Without these restrictions, earlier sentences in the derivation would not semantically entail later sentences
- For QND to be sound, we need $\Gamma \vdash_{QND} \mathcal{P}$ to be sufficient for $\Gamma \vDash \mathcal{P}$.
- As with SND, we will prove this by showing that the set of open assumptions Γ_k on line $\#k$ semantically entail the sentence $\mathcal{P}_k$ on that line, for all lines k in any QND derivation

13. Metalogic for QL

d. Soundness of System QND

Semantic entailment for infinitely-many premises

- Let Γ be a possibly infinite set of QL-**sentences**; Θ a conclusion
- An argument is **semantically invalid** if there is a model $\mathfrak{M}$ that makes true each sentence in Γ but which makes Θ false
- In this case we write $\Gamma \nvDash \Theta$
- If there is no such QL-model, then $\Gamma \vDash \Theta$, i.e. if whenever we have $\mathfrak{M} \vDash \Gamma$ we also have $\mathfrak{M} \vDash \Theta$

QND derivability for infinitely-many premises

- Θ is **QND-derivable** from Γ provided there is an QND derivation:
  1.) whose starting premises Δ are a finite subset of Γ

  2.) in which Θ appears on its own in the final line

  3.) where Θ is directly next to the main scope line, i.e. only in the scope of the Δ-premises
- In this case, we write $\Gamma \vdash_{QND} \Theta$ (also: $\Delta \vdash_{QND} \Theta$)
- If no such derivation exists, then we say that Θ is NOT QND-derivable from Γ, and we write $\Gamma \nvdash_{QND} \Theta$

Soundness: Proof Idea and notation

- Subgoal: given any line in a QND derivation, show that the QL–*sentence* on that line is entailed by the premises or assumptions accessible from that line
- Let "P_k" be the sentence on line k of our derivation
- Let "Γ_k" be the set of premises/assumptions accessible on line k, i.e. the set of open assumptions/premises in whose scope P_k lies
- **Subgoal**: given a sentence P_k on line k, show that $\Gamma_k \vDash P_k$

Soundness: Proof Strategy

- Recall that QND derivations are defined recursively: from a (possibly empty) set of premises, we have a finite number of rules to add a line

  – These ways include all our SND rules plus an intro and elimination rule for our quantifiers $\forall$ and $\exists$
- Hence: do induction on the number of lines in an QND derivation
- Show that the base case has the property (line #1)
- Induction hypothesis: assume the property holds for all lines $\leq k$.
- Induction step: show the property holds for line #k+1 (by considering all possible ways line #k+1 could arise)

Let's remain Righteous!

- Recall: a line *i* of a derivation is **righteous** just in case $\Gamma_i \vDash P_i$, i.e. just in case **the set of assumptions/premises accessible from** ***i*** semantically entail the sentence on that line.
- Call a derivation *righteous* if every line in it is righteous
- Our goal is to prove that every derivation in QND is righteous!
- We will extend our induction for SND to cover our four new rules!

From righteousness to soundness:

- Let Γ be any set of QL sentences (possibly infinite)
- If $\Gamma \vdash_{QND} \mathcal{P}$, then by definition there is a derivation from finitely-many premises $\Delta \subseteq \Gamma$, such that $\mathcal{P}$ occurs on the final line and lies in the scope of Δ (i.e. $\Delta \vdash_{QND} \mathcal{P}$)
- Then by righteousness, $\Delta \vDash \mathcal{P}$

  - i.e. any model $\mathfrak{M}$ that makes Δ true must make $\mathcal{P}$ true
- So there is no QL-model that makes all the sentences in Γ true while making $\mathcal{P}$ false, so $\Gamma \vDash \mathcal{P}$ as well
- So we will have shown **Soundness**: If $\Gamma \vdash_{QND} \mathcal{P}$, then $\Gamma \vDash \mathcal{P}$

Base Case

- **Base case**: for any QND derivation, show that $\Gamma_1 \vDash \mathcal{P}_1$.
- Proof: Γ_1 is the set of premises accessible at line #1, which comprises exactly the QL–sentence $\mathcal{P}_1$
- (recall that every premise of a derivation lies in its own scope)
- Clearly, $\mathcal{P}_1 \vDash \mathcal{P}_1$, so $\{\mathcal{P}_1\} \vDash \mathcal{P}_1$
- So line #1 is righteous (i.e. $\Gamma_1 \vDash \mathcal{P}_1$)

Stating the Induction Step

- **Induction Hypothesis**: Assume that every line i for $1 < i \leq k$ is righteous (i.e. that $\Gamma_i \vDash \mathcal{P}_i$)

- Induction step: Consider line #k+1; show that $\Gamma_{k+1} \vDash \mathcal{P}_{k+1}$

- We have 16 cases to consider! We have essentially already considered 12 of these from our soundness proof for SND

- We have four new cases: our intro. and elimin. rules for $\forall$ and $\exists$

Cases 1–12: modifying our soundness proof for SND

- For each of these 12 cases, we simply replace “truth-value assignments” with “QL-models” (or interpretations), along with replacing truth-functional semantic notions with ones defined for quantifier logic
- e.g. quantificational entailment, quantificational consistency/satisfiability
- e.g. “$\Gamma_{k+1} \vDash \mathcal{P}_{k+1}$” now means “sentence $\mathcal{P}_{k+1}$ is true in all models that make-true the premise set Γ_{k+1}”.
- Equivalently: $\Gamma_{k+1} \cup \{\sim\mathcal{P}_{k+1}\}$ is unsatisfiable in QL

Case 13 (gasp): For-all Elimination

- **Case 13**: $\mathcal{P}_{k+1}$ is derived by Universal Elimination (:$\forall E$)
  Show that $\Gamma_{k+1} \vDash \mathcal{P}_{k+1}$
- $\mathcal{P}_{k+1}$ must have the form $\mathcal{Q}[c/\chi]$ (read "c for χ")
- By the IH, line #h is righteous, so $\Gamma_h \vDash (\forall \chi)\mathcal{Q}$
- Since every assumption that is accessible at line #h is also accessible at line #k+1, we have $\Gamma_h \subseteq \Gamma_{k+1}$
- Hence, $\Gamma_{k+1} \vDash (\forall \chi)\mathcal{Q}$
- **Lemma**: a universally quantified sentence entails each of its substitution instances.
  $\Rightarrow$ any model that makes-true $(\forall \chi)\mathcal{Q}$ also makes-true $\mathcal{Q}[c/\chi]$
- Hence, $\Gamma_{k+1} \vDash \mathcal{Q}[c/\chi]$

Lemma for Case 13

- **Lemma**: a universally quantified sentence entails each of its substitution instances: $(\forall \chi)\mathcal{Q} \vDash \mathcal{Q}[c/\chi]$ for each constant c
- Consider an arbitrary model $\mathfrak{M}$ that makes true $(\forall \chi)\mathcal{Q}$
- Then by defN of true-in-QL, there is some variable assignment d_I that satisfies $(\forall \chi)\mathcal{Q}$ in $\mathfrak{M}$
- By the satisfaction-conditions for univ. quant. sentences, this means that for each $r \in D$, the variant $d_I[r/x]$ satisfies $\mathcal{Q}$
- So for each c, I must assign c an object $r \in D$ s.t. $d_I[I(c)/x]$ satisfies $\mathcal{Q}$.
- Hence, variable assignment d_I satisfies $\mathcal{Q}[c/\chi]$ (Lemma 11.1.1)
- (Intuition: no matter which r in D is assigned to c, $\mathcal{Q}[c/\chi]$ is true)
- So, for each constant c, $\mathfrak{M} \vDash \mathcal{Q}[c/\chi]$, i.e. is true in the model

Case 14: Existential Introduction

- **Case 14**: $\mathcal{P}_{k+1}$ is derived by Existential Introduction ($:\exists I$)
- $\mathcal{P}_{k+1}$ must have the form $(\exists \chi)\mathcal{Q}\lceil \chi/c \rceil$ (read "χ for some c")
- By the IH, line #h is righteous, so $\Gamma_h \vDash \mathcal{Q}$ (where c appears)
- Since every assumption that is accessible at line #h is also accessible at line #k+1, we have $\Gamma_h \subseteq \Gamma_{k+1}$
- Hence, $\Gamma_{k+1} \vDash \mathcal{Q}$
- **Lemma**: a sentence $\mathcal{Q}$ entails any existentially quantified (possibly partial) substitution instance $(\exists \chi)\mathcal{Q}\lceil \chi/c \rceil$.
- Hence, $\Gamma_{k+1} \vDash (\exists \chi)\mathcal{Q}\lceil \chi/c \rceil$

Lemma for Case 14

- **Lemma**: a sentence $\mathcal{Q}$ entails any existentially quantified (possibly partial) substitution instance $(\exists \chi)\mathcal{Q}\lceil \chi / c \rceil$
- Consider an arbitrary model $\mathfrak{M}$ that makes true $\mathcal{Q}[c]$
- Then by defN of true-in-QL, there is some variable assignment d_I that satisfies $\mathcal{Q}[c]$ in $\mathfrak{M}$. Let r be the object in D that c stands for.
- Then $d_I[r/\chi]$ satisfies $\mathcal{Q}\lceil \chi / c \rceil$ (i.e. the open sentence we get by replacing some c's with χ is satisfied by object r)
- Recall: d_I satisfies $(\exists \chi)\mathcal{Q}\lceil \chi / c \rceil$ provided there is some object $r \in D$ s.t. $d_I[r/\chi]$ satisfies $\mathcal{Q}\lceil \chi / c \rceil$
- Hence, by the defN of true-in-QL, $(\exists \chi)\mathcal{Q}\lceil \chi / c \rceil$ is true in $\mathfrak{M}$

Case 16: Existential Elimination

- **Case 16**: $\mathcal{P}_{k+1}$ is derived by Existential Elimination ($:\exists E$)
- We require that c not occur in $(\exists \chi)\mathcal{Q}$, $\mathcal{P}_{k+1}$, or Γ_{k+1}
- By the IH, lines #h and #m are righteous, so $\Gamma_h \vDash (\exists \chi)\mathcal{Q}$ and $\Gamma_m \vDash \mathcal{P}_{k+1}$
- Since every assumption/premise that is accessible at line #h is also accessible at line #k+1, we have $\Gamma_h \subseteq \Gamma_{k+1}$. So $\Gamma_{k+1} \vDash (\exists \chi)\mathcal{Q}$
- Note that every member of Γ_m is accessible at #k+1 except assumption $\mathcal{Q}[c/\chi]$ on line #j. So $\Gamma_m \subseteq \Gamma_{k+1} \cup \{\mathcal{Q}[c/\chi]\}$
- So since $\Gamma_m \vDash \mathcal{P}_{k+1}$, we have $\Gamma_{k+1} \cup \{\mathcal{Q}[c/\chi]\} \vDash \mathcal{P}_{k+1}$
- **Lemma**: if (i) constant c does not occur in $(\exists \chi)\mathcal{Q}$, $\mathcal{P}$, or set Γ, (ii) $\Gamma \vDash (\exists \chi)\mathcal{Q}$ and (iii) $\Gamma \cup \{\mathcal{Q}[c/\chi]\} \vDash \mathcal{P}$, then $\Gamma \vDash \mathcal{P}$
- Hence, $\Gamma_{k+1} \vDash \mathcal{P}_{k+1}$, so line #k+1 is righteous!

Locality Lemma (Book's 11.1.7)

- **Locality Lemma**: 'local agreement' on interpretation between models arises iff there is 'agreement' on entailment relations.
- Set-up (for a given sentence $\mathcal{P}$): Consider two QL-models $\mathfrak{M}^1 := (D, I_1)$ and $\mathfrak{M}^2 := (D, I_2)$ with the same domain D, whose interpretation functions I_1 and I_2 give the same interpretations for any constants or predicates appearing in QL-sentence $\mathcal{P}$

  (so any differences between $\mathfrak{M}^1$ and $\mathfrak{M}^2$ arise from how they interpret QL-symbols NOT appearing in $\mathcal{P}$).
- Then $\mathfrak{M}^1 \vDash \mathcal{P}$ if and only if $\mathfrak{M}^2 \vDash \mathcal{P}$
- We will use this lemma for the cases of $\forall I$ and $\exists E$

Lemma for Case 16

- **Lemma**: if (i) constant c does not occur in $(\exists \mathcal{x})\mathcal{Q}$, $\mathcal{P}$, or set Γ, (ii) $\Gamma \vDash (\exists \mathcal{x})\mathcal{Q}$ and (iii) $\Gamma \cup \{\mathcal{Q}[c/\mathcal{x}]\} \vDash \mathcal{P}$, then $\Gamma \vDash \mathcal{P}$
- NTS: In a model $\mathfrak{M} := (D, I)$ that makes all members of Γ true (i.e. $\mathfrak{M} \vDash \Gamma$), $\mathcal{P}$ is true (i.e. show that $\mathfrak{M} \vDash \mathcal{P}$)
- Since $\Gamma \vDash (\exists \mathcal{x})\mathcal{Q}$, there exists some object $r \in D$ that satisfies $\mathcal{Q}$ (i.e. there exists a d_I s.t. $d_I[r/\mathcal{x}]$ satisfies $\mathcal{Q}$)
- Since c does not occur in $(\exists \mathcal{x})\mathcal{Q}$, $\mathcal{P}$, or set Γ, we can define a new model $\mathfrak{M}'$ that is just like $\mathfrak{M}$ except that $I'(c) = r$.
- Then since $d_I[r/\mathcal{x}]$ satisfies $\mathcal{Q}$, we have $d_{I'}[r/\mathcal{x}] = d_{I'}[I'(c)/\mathcal{x}]$ satisfies $\mathcal{Q}$ as well.
- So by Lemma 11.1.1, $d_{I'}$ satisfies $\mathcal{Q}[c/\mathcal{x}]$, so $\mathfrak{M}' \vDash \mathcal{Q}[c/\mathcal{x}]$
- By Locality, since $\mathfrak{M} \vDash \Gamma$, we have $\mathfrak{M}' \vDash \Gamma$ too
- So $\mathfrak{M}' \vDash \Gamma \cup \{\mathcal{Q}[c/\mathcal{x}]\}$ which by assumption $\vDash \mathcal{P}$. So $\mathfrak{M}' \vDash \mathcal{P}$
- Hence, by Locality, $\mathfrak{M} \vDash \mathcal{P}$, so $\Gamma \vDash \mathcal{P}$

Case 15: For-all Introduction

- **Case 15**: $\mathcal{P}_{k+1}$ is derived by Universal Introduction ($:\forall I$)
- $\mathcal{P}_{k+1}$ must have the form $(\forall\chi)\mathcal{Q}[\chi/c]$, with c not appearing in Γ_h
- By the IH, line #h is righteous, so $\Gamma_h \vDash \mathcal{Q}$ (where c appears)
- Since every assumption/premise that is accessible at line #h is also accessible at line #k+1, we have $\Gamma_h \subseteq \Gamma_{k+1}$
- Hence, $\Gamma_{k+1} \vDash \mathcal{Q}$
- **Lemma**: if c does not appear in any member of set Γ, then if $\Gamma \vDash \mathcal{Q}$, we have $\Gamma \vDash (\forall\chi)\mathcal{Q}[\chi/c]$
- Our rule $\forall I$ requires that c does not appear in Γ_{k+1}, so by the lemma, $\Gamma_{k+1} \vDash (\forall\chi)\mathcal{Q}[\chi/c]$

Lemma for Case 15

- **Lemma**: if c does not appear in any member of set Γ, then if $\Gamma \vDash \mathcal{Q}$, we have $\Gamma \vDash (\forall \chi)\mathcal{Q}[\chi/c]$
- Consider an arbitrary model $\mathfrak{M}$ that makes true all of the sentences in Γ. Then $\mathfrak{M} \vDash \mathcal{Q}$ (i.e. $\mathcal{Q}$ is true in $\mathfrak{M}$)
- So there exists a variable assignment d_I that satisfies $\mathcal{Q}$ in $\mathfrak{M}$
- Goal: show that there exists a variable assignment d'_I that satisfies $(\forall \chi)\mathcal{Q}[\chi/c]$
    - i.e. a d'_I s.t. $d'_I[r/\chi]$ satisfies $\mathcal{Q}[\chi/c]$ for each object $r \in D$
    - We will actually show that the given d_I does the trick!

Lemma for Case 15 continued

- Notice that for $(\forall x)\mathcal{Q}[x/c]$ to be a wff, x must not already occur in sentence $\mathcal{Q}$, so $\mathcal{Q}$ can't already have a x–quantifier.
- So x occurs freely in $\mathcal{Q}[x/c]$ as the **only** free variable (since $(\forall x)\mathcal{Q}[x/c]$ is, by assumption, a sentence)
- Hence, a variable assignment d_I of free variables in $\mathcal{Q}[x/c]$ to objects in D amounts to a choice of object $r \in D$ to assign x
- So d_I must make some choice $r := I(c)$ of object to assign x
- Hence d_I simply equals $d_I[r/x]$.
- By 11.1.1, d_I satisfies $\mathcal{Q}[x/c]$ iff $d_I[I(c)/x]$ satisfies $\mathcal{Q}$
- So d_I and hence $d_I[r/x]$ satisfies $\mathcal{Q}[x/c]$

Lemma for Case 15 continued *more*

- For each $r \in D$, we define a new model $\mathfrak{M}_r$ whose interpretation function I_r is just like I except that it assigns the constant c to r
- Now, $\mathfrak{M} \vDash \Gamma$ and $\mathfrak{M}_r$ differs by $\mathfrak{M}$ only in how it interprets a symbol that does not occur in Γ.
- So by Locality, we have $\mathfrak{M}_r \vDash \Gamma$ and hence $\mathfrak{M}_r \vDash \mathcal{Q}$
- So for each object $r \in D$, we have a $d_{I_r}[r/\chi]$ that satisfies $\mathcal{Q}[\chi/c]$.
- This is equivalent to saying that for each $r \in D$, $d_I[r/\chi]$ satisfies $\mathcal{Q}[\chi/c]$, since d_I and each d_{I_r} agree on what to assign every other variable besides possibly χ
- And by defN, this means that d_I satisfies $(\forall\chi)\mathcal{Q}[\chi/c]$, and hence this sentence is true in $\mathfrak{M}$

13. Completeness of QND

Completeness of QND

- **QND is Complete**: For any set Γ of QL–sentences and any QL–sentence $\mathcal{P}$, if Γ semantically entails $\mathcal{P}$, then there exists a derivation of $\mathcal{P}$ from Γ in our natural deduction system QND
  - In symbols: If $\Gamma \vDash \Theta$, then $\Gamma \vdash_{QND} \Theta$
  - Note that Γ can be countably infinite

- Completeness guarantees that for any valid QL–argument, there is at least one corresponding deduction in QND.
- So we need not reason about arbitrary models to determine if a QL–argument is valid; reasoning in QND suffices! WOW COOL

“$\vDash$”: our Semantic Double Turnstile

- “$\Gamma \vDash \mathcal{P}$” means that Γ logically entails $\mathcal{P}$
  In any QL-model $\mathfrak{M}$ where the premises in Γ are true, the conclusion $\mathcal{P}$ is true
- Equivalently: there is no QL-model $\mathfrak{M}$ such that Γ is satisfied while $\mathcal{P}$ is false
- Equivalently, this means that $\Gamma \cup \{{\sim}\mathcal{P}\}$ **is unsatisfiable**:
  no QL-model makes-true the premises and negated conclusion
- We’ll use this last fact A LOT in our proof that QND is complete!

13. Completeness of QND

a. Semantic vs. Syntactic Consistency

Semantic vs. Syntactic Consistency

- As with SND, we appeal to two distinct notions of consistency
- One is **semantic**:

  there is a QL-model that **satisfies** every sentence in the set
- We introduce a new **syntactic** notion of consistency relative to QND:

  - a set of QL sentences is **QND-consistent** provided that you can't derive contradictory sentences from it in QND
- Core proof idea: we'll show that if a set of sentences is **QND-consistent**, then it is also semantically consistent (i.e. **satisfiable**). So by the contrapositive: if a set is **un**satisfiable, then it is **in**consistent-in-QND.

Semantic: Satisfiable (quantificationally consistent)

- Recall: a set of QL sentences is **satisfiable** provided there is at least one QL-model $\mathfrak{M}$ that makes all of them true
- This is a *semantic* notion of consistency (aka “quantificational consistency”)
- Contrast this with the syntactic notion of **consistency in QND**:

Syntactic: (In)consistent-in-QND (derivationally consistent)

- Let Γ be a (possibly infinite) set of QL sentences
- **Inconsistent-in-QND**: from premises in Γ, we can derive contradictory formulas R and $\sim R$ in the scope of the main scope line (i.e. in the scope of these premises)
- **Consistent-in-QND**: Γ is not QND-inconsistent, i.e. there is no derivation from premises in Γ resulting in contradictory formulas within the main scope
- Other words we might use for these concepts: QND-inconsistent, derivationally-inconsistent, QND-consistent, etc.
- Just remember: this syntactic notion has nothing to do with models or interpretations!

13. Completeness of QND

b. Proof Sketch

Proof Sketch: Just like what we did for SL!

- Goal: prove the completeness of QL: for every QL sentence $\mathcal{P}$ and every set Γ of QL sentences, if $\Gamma \vDash \mathcal{P}$ then $\Gamma \vdash_{QND} \mathcal{P}$
- So assume that $\Gamma \vDash \mathcal{P}$.
- This means that $\Gamma \cup \{\sim\mathcal{P}\}$ is **unsatisfiable**:
  no QL-model satisfies the premises and negated conclusion
  (i.e. $\Gamma \cup \{\sim\mathcal{P}\}$ is *semantically* inconsistent)
- We now appeal to a **Consistency lemma** that remains the heart of the enterprise: any QND-consistent set of QL sentences is satisfiable (i.e. semantically consistent)

Proof Sketch: Using the consistency lemma

- **Consistency lemma** (CL): any QND–consistent set of QL sentences is satisfiable, i.e. true in some QL–model $\mathfrak{M}$
- **Contrapositive** of CL: any set of QL sentences that is Unsatisfiable is QND–Inconsistent
- From $\Gamma \vDash \mathcal{P}$ we know that $\Gamma \cup \{\sim\mathcal{P}\}$ is unsatisfiable
- So by the contrapositive of CL, we see that $\Gamma \cup \{\sim\mathcal{P}\}$ is QND–inconsistent
- This means that we can derive a pair of contradictory sentences R and $\sim R$ from $\Gamma \cup \{\sim\mathcal{P}\}$! So using the power of negation elimination, we can derive $\mathcal{P}$ from Γ, i.e. $\Gamma \vdash_{QND} \mathcal{P}$. So we are 'done'!

Negation Elimination Refresher (book's Exercise 11.4.2)

- Claim: if $\Gamma \cup \{\sim\mathcal{P}\}$ is **QND-inconsistent**, then $\Gamma \vdash_{QND} \mathcal{P}$
- Proof: starting with (finitely-many) premises Δ from Γ, introduce $\sim\mathcal{P}$ as a subproof assumption for negation elimination
- Since $\Gamma \cup \{\sim\mathcal{P}\}$ is QND-inconsistent, we can derive a contradictory pair R and $\sim R$ within the scope of sentences in $\Delta \cup \{\sim\mathcal{P}\}$
- Then discharge this assumption $\sim\mathcal{P}$ by negation elimination, writing $\mathcal{P}$, now in the scope of Δ. So $\Delta \vdash_{QND} \mathcal{P}$
- Since $\Delta \subseteq \Gamma$, we have $\Gamma \vdash_{QND} \mathcal{P}$

Core subgoal: Prove consistency lemma (book's 11.4.2)

- So all we have to do is prove the **consistency lemma**: any QND–consistent set of QL sentences is satisfiable
- As with SL, we'll prove this lemma in several 'stages':
- The first two are straightforward: given a QND–consistent set Γ, we construct a **superset** Γ^* that is *maximally QND–consistent* and *existentially complete* ($\exists$–complete)
- In the third stage, we show that any $\exists$–complete, maximally QND–consistent set is satisfiable: we use maximal consistency and $\exists$–completeness to construct a model that satisfies every sentence in Γ^*. *Wrinkle*: we work in an extended language QL$'$!
- Since by construction $\Gamma \subseteq \Gamma^*$, this QL$'$–model satisfies Γ (in QL$'$)
- From our QL$'$–model, we generate a QL–model that satisfies Γ

13. Completeness of QND

c. Stage 0: ∃-Completeness and QL′

Maximally QND-consistent (no longer enough!)

- A set Γ^* of QL or QL′ sentences is **maximally QND-consistent** provided that:
  1.) Γ^* is QND-consistent (i.e. can't derive contradictory sentences)
  2.) adding **any** additional sentence to Γ^* would result in an QND-inconsistent set
- i.e. for any $P \notin \Gamma^*$, $\{P\} \cup \Gamma^*$ is QND-inconsistent
- Unlike with SL, maximal derivational consistency is no longer enough to ensure satisfiability
- Recall that our purely syntactic membership lemma is motivated by the truth-conditions for QL sentences: sentences belong to Γ^* iff the relevant "truth-condition pieces" belong to Γ^* as well
- To extend our membership lemma to quantified sentences, we require that every existential sentence in Γ^* has a substitution instance also in Γ^*. So we introduce a new property:

Existential-completeness: definition and motivation

- **∃-completeness**: a set Γ of QL or QL$'$ sentences is *existentially-complete* just in case for every sentence in Γ of the form $(\exists \chi)\mathcal{P}$, at least one substitution instance $\mathcal{P}[c/\chi]$ is in Γ

- Motivation: $(\exists \chi)\mathcal{P}$ is true in a model iff some object $r \in D$ is a $\mathcal{P}$
- To construct an ∃-complete set Γ^*, we need recourse to a countable infinity of unused constants.
- Otherwise, new substitution instances that we add could "contradict" sentences already in Γ, spoiling QND-consistency
- *Problem*: our starting Γ might be infinite and so already use infinitely-many constants from QL. What are we to do?

It’s a bird! It’s a plane! It’s ... Language QL′???

- QL′ is exactly like QL except that we allow subscripted *constants* to have **primed-indices**
- e.g. $c_{11'}$, $b_{234'}$, $g_{2'}$ (‘$'$’-symbol always at the end)
- Unsubscripted constants remain the same: *a* thru *v*
- So QL′ just adds one new symbol ‘$'$’, allowed to occur only at the end of indices for constants
- The recursive structure of truth-in-QL′ is defined exactly the same as for QL (using our good friend, satisfaction semantics!)
- Note that we do NOT allow primed indices on Predicates
- Moral: reach *for the stars*, **not** drugs

13. Completeness of QND

d. Stage 1: Constructing Γ^*

Stage 1(i): first enumerate the sentences of QL′!

- Let Γ be a QND-consistent set of QL sentences (possibly infinite)
- To construct Γ^*, we first **enumerate** the QL′ sentences, so that every QL′ sentence is associated with a unique positive integer $\{1, 2, 3, \ldots\}$
- As with SL, stipulate an ‘alphabetical order’ for QL′ symbols
- $\sim, \vee, \&, \supset, \equiv, (,), 0, 1, \ldots, 9, A, B, \ldots, Z, a, \ldots, v, w, x, y, z, \forall, \exists, '$
- Assign each symbol an **index** between ‘10’ and ‘84’ (skip 17-19)
- Then each QL′ sentence corresponds to a unique positive integer, constructed by replacing each symbol in the sentence with its index, from left to right.
- So with our ordering, ‘A’ is the first sentence; ‘B’ the second . . . up to Z, and then we hit $\sim A$ ($\mapsto$ 1030), then $\sim B$ ($\mapsto$ 1031), etc.

Recall what we did in SL to form Γ^* Max.–SND–Consist.

- We considered the first sentence 'A' in our enumeration. If A could be added to Γ without the resulting set being SND–inconsistent, then we let $\Gamma_1 := \Gamma \cup \{A\}$.
- Otherwise, let $\Gamma_1 := \Gamma$ (so that Γ_1 stays SND–consistent)
- We proceeded to the 2nd sentence in our enumeration. If it could be added to Γ_1 without the new set being SND–inconsistent, let Γ_2 be the result. Otherwise, let $\Gamma_2 := \Gamma_1$
- Γ^* was the result of 'doing' this procedure for every SL sentence
- Now we need to complicate matters a bit, to handle sentences of the form $(\exists\chi)\mathcal{P}$ and ensure we add a suitable substitution instance whenever we can add $(\exists\chi)\mathcal{P}$ while preserving QND–consistency

Building up Γ^*

- Given a QND-consistent set of QL sentences Γ, let $\Gamma_0 := \Gamma$
- Consider the k-th sentence P_k in our enumeration of QL$'$
- Define Γ_{k+1} as follows:
  i.) Γ_k if the set $\Gamma_k \cup \{P_k\}$ is QND-**INconsistent**
  ii.) $\Gamma_k \cup \{P_k\}$ if P_k does NOT have the form $(\exists \mathcal{X})\mathcal{Q}$, and $\Gamma_k \cup \{P_k\}$ is QND-consistent
  iii.) $\Gamma_k \cup \{P_k, P_k^\dagger\}$ if $\Gamma_k \cup \{P_k\}$ is QND-consistent AND P_k DOES have the form $(\exists \mathcal{X})\mathcal{Q}$, where $P_k^\dagger$ is a substitution instance $\mathcal{Q}[c/\mathcal{X}]$, and c is the alphabetically earliest constant not in P_k or any sentence in Γ_k
  - Such a c is guaranteed to exist because Γ_0 belongs to QL.
  - So the countable-infinity of primed subscripted constants from QL$'$ are available at each stage if needed.
- Then $\Gamma^* := \bigcup_{k=0}^{\infty} \Gamma_k$

13. Completeness of QND

e. Stage 2: Γ^* is M-QND-C & $\exists$-complete

Stage 2: Γ^* is maximally QND-consistent & $\exists$-complete

- This requires proving three claims (from the definitions):

  1.) Γ^* is consistent in QND

  2.) Adding any additional sentence to Γ^* would result in a **QND-inconsistent** set

  3.) For every QL′ sentence of the form $(\exists \chi)\mathcal{Q}$ in Γ^*, at least one substitution instance $\mathcal{Q}[c/\chi]$ belongs to Γ^*

- We prove these in turn

Stage 2 (i): Γ^* is QND-consistent

- Assume for *reductio* that Γ^* is inconsistent in QND
- Then there would be a QND derivation with finite premise set $\Delta \subset \Gamma^*$ that derives a contradictory pair R and $\sim R$
- Since Δ is finite, there exists some $k+1 \in \mathbb{N}$ s.t. $\Delta \subset \Gamma_{k+1}$. So then this Γ_{k+1} would be QND-inconsistent.
- Yet, each Γ_{k+1} is necessarily QND-consistent:
  - If P_k is not existential, it joins Γ_{k+1} only if $\Gamma_k \cup \{P_k\}$ is QND-consistent—by condition (ii)
  - If P_k is of the form $(\exists \mathcal{X})\mathcal{Q}$, it joins only if $\Gamma_k \cup \{P_k\}$ is QND-consistent.
    - It remains to show that $\Gamma_k \cup \{(\exists \mathcal{X})\mathcal{Q}, \mathcal{Q}[c/\mathcal{X}]\}$ is QND-consistent
  - **Lemma**: if c does not occur in a QND-C set $\Gamma_k \cup \{(\exists \mathcal{X})\mathcal{Q}\}$, then $\Gamma_k \cup \{(\exists \mathcal{X})\mathcal{Q}, \mathcal{Q}[c/\mathcal{X}]\}$ is QND-consistent
- Hence, Γ^* must be QND-consistent, on pain of *reductio*

Stage 2 (ii): Γ^* is maximally QND-consistent

- Assume for *reductio* that Γ^* weren't maximally QND-consistent, despite being QND-consistent
- i.e. assume *it is not the case that* for all other sentences, adding it to Γ^* would result in a QND-inconsistent set
  $\Rightarrow$ there exists a sentence $\mathcal{Q}$ that we could add to Γ^* while preserving QND-consistency (i.e. there is some sentence we neglected that could make Γ^* a 'bigger' QND-consistent set)
- Yet, $\mathcal{Q}$ would appear in our enumeration as some sentence P_k, 'considered' at the k-th stage of our construction of Γ^*.
- So if $\mathcal{Q}$ isn't in Γ^*, then this is because adding it 'would have' made $\Gamma_k \subset \Gamma^*$ QND-inconsistent.
  So $\{\mathcal{Q}\} \cup \Gamma^*$ must be QND-inconsistent (*reductio*!)
- So we can't add any $\mathcal{Q}$ to Γ^* while preserving QND-consistency

Stage 2 (iii): Γ^* is $\exists$–complete

- We simply need to show that for each sentence of the form $(\exists \mathcal{X})\mathcal{Q} \in \Gamma^*$, a substitution instance $\mathcal{Q}[c/\mathcal{X}]$ also belongs to Γ^*
- Note that this is true by construction: each sentence of the form $(\exists \mathcal{X})\mathcal{Q}$ occurs in our QL′–enumeration:
- If we could have "added" $(\exists \mathcal{X})\mathcal{Q}$ at the k–th stage while preserving QND–consistency, then we also added a substitution instance.
- This is so even if $(\exists \mathcal{X})\mathcal{Q}$ is already in $\Gamma_0 := \Gamma$, since by condition (iii) $\Gamma_{k+1} := \Gamma_k \cup \{(\exists \mathcal{X})\mathcal{Q}, \mathcal{Q}[c/\mathcal{X}]\}$ which in this case would equal $\Gamma_k \cup \{\mathcal{Q}[c/\mathcal{X}]\}$ (since in this case, $(\exists \mathcal{X})\mathcal{Q} \in \Gamma_k$)

13. Completeness of QND

f. Stage 3: Model Construction

Stage 3: The Maximal Consistency Lemma ($\approx$ book's 11.4.7)

- **∃–C Maximal Consistency Lemma**: every QL$'$ set that is maximally–QND–consistent and ∃–complete is satisfiable
- (there exists a QL$'$–model that makes–true every sentence in Γ^*)
  We construct this model, calling it "$\mathfrak{M}^*$" ($\approx$book's "$\mathbf{I}^*$")
- Proof idea: since Γ^* is M–QND–C, for any sentence $\mathcal{P}$, either $\mathcal{P} \in \Gamma^*$ or $\sim\mathcal{P} \in \Gamma^*$ (you're either in the club or your 'nemesis' is!)

  This holds in particular for each QL$'$–atomic sentence
- Construct a QL$'$–model $\mathfrak{M}^*$ such that for each atomic QL$'$–sentence $\mathcal{A}$, $\mathfrak{M}^* \vDash \mathcal{A}$ iff $\mathcal{A} \in \Gamma^*$
- Then by the recursive structure of QL$'$ sentences, $\mathfrak{M}^* \vDash \mathcal{P}$ iff $\mathcal{P} \in \Gamma^*$

Stage 3 (i): the Membership Lemma (book's 11.4.6)

- To induct on QL′, we first constrain Γ^* membership
- Basically, Γ^* is *THE* club with the MOST ANGELIC bouncer you've eva seen, who enforces maximal consistency. Before this ⁎angel⁎ lets a sentence into Γ^*, he checks who else is GOOD. You hear?
- **Membership Lemma** for club: if $\mathcal{P}$ and $\mathcal{Q}$ are QL′ sentences, then:
  a.) $\sim\mathcal{P} \in \Gamma^*$ if and only if $\mathcal{P} \notin \Gamma^*$
  b.) $\mathcal{P} \,\&\, \mathcal{Q} \in \Gamma^*$ if and only if both $\mathcal{P} \in \Gamma^*$ and $\mathcal{Q} \in \Gamma^*$
  c.) $\mathcal{P} \vee \mathcal{Q} \in \Gamma^*$ if and only if either $\mathcal{P} \in \Gamma^*$ or $\mathcal{Q} \in \Gamma^*$
  d.) $\mathcal{P} \supset \mathcal{Q} \in \Gamma^*$ if and only if either $\mathcal{P} \notin \Gamma^*$ or $\mathcal{Q} \in \Gamma^*$
  e.) $\mathcal{P} \equiv \mathcal{Q} \in \Gamma^*$ iff either (i) $\mathcal{P} \in \Gamma^*$ and $\mathcal{Q} \in \Gamma^*$ or (ii) $\mathcal{P} \notin \Gamma^*$ and $\mathcal{Q} \notin \Gamma^*$
  f.) $(\forall \chi)\mathcal{P} \in \Gamma^*$ iff for each constant c, $\mathcal{P}[c/\chi] \in \Gamma^*$
  g.) $(\exists \chi)\mathcal{P} \in \Gamma^*$ iff for at least one constant c, $\mathcal{P}[c/\chi] \in \Gamma^*$

Stage 3 (i): The Stairway to heaven (book's 11.4.5)

- To prove the membership lemma's cases (a)-(g), we'll use another lemma (*NB*: and she's buying, a lemma, to heavennnnnnnnn!):
- **The Stairway**: if $\Gamma \vdash P$, and Γ^* is a maximally QND-consistent superset of Γ, then $P \in \Gamma^*$ (mnemonic: "$\Gamma \vdash P$" pushes P up to QL′-heaven!)
- Proof: first, assume that $\Gamma \vdash P$ (we'll use this fact below)
  - Next, assume for *reductio* that $P \notin \Gamma^*$. Then since Γ^* is maximally QND-consistent, $\Gamma^* \cup \{P\}$ must be inconsistent in QND.
  - Hence, by negation introduction, $\Gamma^* \vdash \sim P$
  - By assumption, $\Gamma \vdash P$, so also $\Gamma^* \vdash P$, since $\Gamma \subseteq \Gamma^*$
  - So Γ^* derives both P and $\sim P$. *Reductio*! (since Γ^* is M-QND-C)
  - Hence, if $\Gamma \vdash P$ and $\Gamma \subseteq \Gamma^*$, then P must belong to Γ^*

Membership Lemma: Cases (a)–(e)

- I have a feeling that. . .
- WE'VE SEEN THIS incredible content BEFORE! (for SL)
- see the next slide for a refresher
- *Long story short*: There's a feeling I get

  When I look to the west

  And my spirit is crying for leaving

Membership Lemma: Case (a)

- **Case (a)**: $\sim\mathcal{P} \in \Gamma^*$ if and only if $\mathcal{P} \notin \Gamma^*$
- Two directions to prove:

  $\Rightarrow$: Assume $\sim\mathcal{P} \in \Gamma^*$. Then if $\mathcal{P}$ were in Γ^*, we could derive contradictory sentences.

  So since Γ^* is QND–consistent, we must have $\mathcal{P} \notin \Gamma^*$

  $\Leftarrow$: Assume $\mathcal{P} \notin \Gamma^*$. Then adding $\mathcal{P}$ to Γ^* results in an QND–inconsistent set. Hence, there is some finite subset $\Delta \subset \Gamma^*$ s.t. $\Delta \cup \{\mathcal{P}\}$ is QND–inconsistent (i.e. derives contradictory sentence pair).
- So by negation introduction, $\Delta \vdash \sim\mathcal{P}$
- So by **The Stairway**, $\sim\mathcal{P} \in \Gamma^*$

Membership Lemma: Case (f) (something Universally new)

- **Case (f)**: $(\forall \chi)\mathcal{P} \in \Gamma^*$ iff for each constant c, $\mathcal{P}[c/\chi] \in \Gamma^*$
- Two directions to prove:

  $\Rightarrow$: Assume $(\forall \chi)\mathcal{P} \in \Gamma^*$

  - Then for any substitution instance $\mathcal{P}[c/\chi]$, we note that $(\forall \chi)\mathcal{P} \vdash_{QND} \mathcal{P}[c/\chi]$ by $\forall E$. So by the Stairway, $\mathcal{P}[c/\chi] \in \Gamma^*$

  $\Leftarrow$: Assume $(\forall \chi)\mathcal{P} \notin \Gamma^*$. Show that for some constant c, $\mathcal{P}[c/\chi] \notin \Gamma^*$

  - Then $\sim(\forall \chi)\mathcal{P} \in \Gamma^*$ by membership clause (a)

  - Then the derivation on p. 573 or—if I have no life—the derivation on the next slide, shows by the Stairway that $(\exists \chi)\sim\mathcal{P} \in \Gamma^*$, i.e. $\sim(\forall \chi)\mathcal{P} \vdash_{QND} (\exists \chi)\sim\mathcal{P}$

  - Then since Γ^* is $\exists$–complete, there is at least one substitution instance $\sim\mathcal{P}[b/\chi] \in \Gamma^*$. So by (a), $\mathcal{P}[b/\chi] \notin \Gamma^*$, which is what we needed to show.

Membership Lemma: Case (g) (it's getting existential)

- **Case (g)**: $(\exists \mathcal{x})\mathcal{P} \in \Gamma^*$ iff for at least one constant c, $\mathcal{P}[c/\mathcal{x}] \in \Gamma^*$
- $\Rightarrow$: Assume $(\exists \mathcal{x})\mathcal{P} \in \Gamma^*$

  Then since Γ^* is $\exists$–complete, there is at least one substitution instance $\mathcal{P}[c/\mathcal{x}] \in \Gamma^*$
- $\Leftarrow$: assume that $\mathcal{P}[c/\mathcal{x}] \in \Gamma^*$.

  Note that $\mathcal{P}[c/\mathcal{x}] \vdash_{QND} (\exists \mathcal{x})\mathcal{P}$ by Existential introduction

  So by the Stairway, $(\exists \mathcal{x})\mathcal{P} \in \Gamma^*$
- This completes the Membership Lemma, so we proceed to construct a model that satisfies Γ^* (in virtue of being maximally–QND–consistent and $\exists$–complete)!

Stage 3 (ii): Model construction (smart choices=lazy choices)

- A model's domain can be *any* set of objects. Note that, conveniently, symbols *are* objects ("words are labels on boxes")
- We define $\mathfrak{M}^* := (D, I^*)$ as follows:
  1. Let D = the set of constant symbols in QL′, which includes all QL–constants (e.g. unprimed subscripted constants like j_{22})
  2. For the 0–th place predicates, i.e. the sentence letters B, $I^*(B) = \mathit{true}$ iff $B \in \Gamma^*$
  3. For each QL′–constant c, define $I^*(c) = c$ (each names itself)
  4. For each k–place predicate P, $I^*(P) := \mathit{Ext}(P)$ includes all and only those k–tuples $\langle c_1, \ldots, c_k \rangle$ such that $Pc_1 \ldots c_k \in \Gamma^*$

Some important properties of our Model $\mathfrak{M}^*$

- By condition 3, each individual constant refers to a *unique* member of the domain, namely 'itself' (now 'objectified' in D!)
- For each atomic sentence $\mathcal{A}$ of QL$'$, $\mathfrak{M}^* \vDash \mathcal{A}$ iff $\mathcal{A} \in \Gamma^*$ (follows from conditions 2-4)
- By condition 3, every member of the domain is named by a constant, namely itself
- We will occasionally rely on these properties in our induction

Stage 3 (iii): Induction on QL$'$ (i.e. we still be clubbin')

- Goal: construct a QL$'$–model $\mathfrak{M}^*$ that satisfies the ∃–C M–QND–C set Γ^*, i.e. that makes true everything in Γ^* ($\mathfrak{M}^* \vDash \Gamma^*$)
  - Suffices to construct $\mathfrak{M}^*$ s.t. $\mathfrak{M}^* \vDash \mathcal{P}$ iff $\mathcal{P} \in \Gamma^*$
  Say that a sentence is "**clubbin'** " whenever it meets this property
- We induct on the number of logical operators in a QL$'$ sentence: i.e. the five connectives and two quantifiers ("conquans")
- **Base case**: show that each QL$'$–sentence with zero logical operators is clubbin' (i.e. the QL$'$–atomics be clubbin')
- (Strong) **Induction hypothesis**: assume every QL$'$ sentence with 1 to k–many operators is clubbin'
- Induction step: show that an arbitrary QL$'$ sentence with k+1–many operators is clubbin'

Base Case (true by construction)

- Consider an arbitrary QL′−sentence $\mathcal{A}$ that has zero logical operators. (Two directions to show! “iff”)
- Then $\mathcal{A}$ is either an atomic sentence letter B or of the form $P c_1 \ldots c_n$ for n−place predicate P.
- If a sentence letter, then by part 2 of our definition of $\mathfrak{M}^*$, $I^*(B) = \mathit{true}$ iff $B \in \Gamma^*$ (i.e. $\mathfrak{M}^* \vDash B$ iff $B \in \Gamma^*$)
- If $\mathcal{A}$ is of form $P c_1 \ldots c_n$, then by definition $\mathfrak{M}^* \vDash P c_1 \ldots c_n$ iff $\langle c_1^D, \ldots, c_n^D \rangle \in Ext(P)$.
  By part 4, $\langle c_1^D, \ldots, c_n^D \rangle = \langle c_1, \ldots, c_n \rangle \in Ext(P)$ iff $P c_1 \ldots c_n \in \Gamma^*$
- We proceed to do induction using our QL′ induction schema: an arbitrary sentence $\mathcal{P}$ with k+1−many connectives has one of seven forms, coming from our seven operators

Induction on QL$'$: Cases 1–5

- Cases 1-5 are just like what did to prove the completeness of SND
- See the next slide for a refresher (*mutatis mutandis*)!
- Need to show: $\mathcal{P}$ be clubbin', i.e. $\mathcal{P}$ is true on $\mathfrak{M}^*$ iff $\mathcal{P} \in \Gamma^*$, where $\mathcal{P}$ is arbitrary QL$'$ sentence with k+1−many operators
- Induction hypothesis: assume every QL sentence with 1 to k−many operators is clubbin'
- Case 1: $\mathcal{P}$ has the form $\sim\mathcal{Q}$
- Case 2: $\mathcal{P}$ has the form $\mathcal{Q} \,\&\, \mathcal{R}$
- Case 3: $\mathcal{P}$ has the form $\mathcal{Q} \vee \mathcal{R}$
- Case 4: $\mathcal{P}$ has the form $\mathcal{Q} \supset \mathcal{R}$
- Case 5: $\mathcal{P}$ has the form $\mathcal{Q} \equiv \mathcal{R}$

Induction on QL′: Case 1

- **Case 1**: $\mathcal{P}$ has the form $\sim\mathcal{Q}$, where since $\mathcal{Q}$ has k-operators, it is clubbin by the IH (i.e. $\mathfrak{M}^* \vDash \mathcal{Q}$ if and only if $\mathcal{Q} \in \Gamma^*$)
- NTS: (i) (the $\Rightarrow$direction) if $\mathfrak{M}^* \vDash \mathcal{P}$ then $\mathcal{P} \in \Gamma^*$ and (ii) (the $\Leftarrow$direction) if $\mathcal{P} \in \Gamma^*$, then $\mathfrak{M}^* \vDash \mathcal{P}$

  (*Alternative (ii)*: show contrapositive: if $\mathfrak{M}^* \nvDash \mathcal{P}$, then $\mathcal{P} \notin \Gamma^*$)

$\Rightarrow$ if $\mathfrak{M}^* \vDash \mathcal{P}$, then $\mathfrak{M}^* \nvDash \mathcal{Q}$. Since $\mathcal{Q}$ is clubbin', we have $\mathcal{Q} \notin \Gamma^*$

By Membership lemma (a), $\sim\mathcal{Q} \in \Gamma^*$, so $\mathcal{P} \in \Gamma^*$

$\Leftarrow$ if $\mathcal{P} \in \Gamma^*$, then $\sim\mathcal{Q} \in \Gamma^*$. So by Membership lemma (a), $\mathcal{Q} \notin \Gamma^*$.

Since $\mathcal{Q}$ is clubbin', we have $\mathfrak{M}^* \nvDash \mathcal{Q}$. (i.e. $\mathcal{Q}$ is false in $\mathfrak{M}^*$)

So by the truth conditions for negation, $\mathcal{P}$ is true in $\mathfrak{M}^*$, i.e. $\mathfrak{M}^* \vDash \mathcal{P}$

Induction on QL$'$: Case 7 (existential quantifier)

- **Case 7**: $\mathcal{P}$ has the form $(\exists \chi)\mathcal{Q}$
  (warning: “$\mathcal{Q}$” is not a sentence, so sadly it can’t be clubbin’)
- We will use Membership Lemma **Case (g)**:
  $(\exists \chi)\mathcal{Q} \in \Gamma^*$ iff for at least one constant c, $\mathcal{Q}[c/\chi] \in \Gamma^*$

$\Rightarrow$ Assume $\mathfrak{M}^* \vDash (\exists \chi)\mathcal{Q}$. (need to show that $(\exists \chi)\mathcal{Q} \in \Gamma^*$)
- Then by the truth-conditions for existential, there is some object $r \in D$ that satisfies $\mathcal{Q}$.
- ‘r’ names object r, so substitution instance $\mathcal{Q}[r/\chi]$ is true in $\mathfrak{M}^*$
- This substitution instance has less than $k+1$-operators, so it is clubbin’. Hence, by the IH, $\mathcal{Q}[r/\chi] \in \Gamma^*$ (since $\mathfrak{M}^* \vDash \mathcal{Q}[r/\chi]$)
- So by membership case (g), $(\exists \chi)\mathcal{Q} \in \Gamma^*$

Induction on QL′: Case 7 backwards direction

- **Case 7**: $\mathcal{P}$ has the form $(\exists \chi)\mathcal{Q}$
- Use Membership Lemma **Case (g)**:
  $(\exists \chi)\mathcal{Q} \in \Gamma^*$ iff for at least one constant c, $\mathcal{Q}[c/\chi] \in \Gamma^*$

$\Leftarrow$ Assume $(\exists \chi)\mathcal{Q} \in \Gamma^*$. Show that $\mathfrak{M}^* \vDash (\exists \chi)\mathcal{Q}$
- Then by membership case (g), there is at least one substitution instance $\mathcal{Q}[c/\chi] \in \Gamma^*$, for some constant c
- Since $\mathcal{Q}[c/\chi]$ has fewer than $k+1$–operators, it is clubbin'.
- So by the Induction Hypothesis, $\mathfrak{M}^* \vDash \mathcal{Q}[c/\chi]$.
- Since 'c' names object c, we see that c satisfies $\mathcal{Q}$ in $\mathfrak{M}^*$
- So by the truth–conditions for existentials, $(\exists \chi)\mathcal{Q}$ is true in $\mathfrak{M}^*$

Induction on QL$'$: Case 6 (universal quantifier)

- **Case 6**: $\mathcal{P}$ has the form $(\forall \chi)\mathcal{Q}$
  (warning: "$\mathcal{Q}$" is not a sentence, so it can't be clubbin')
- We will use Membership Lemma **Case (f)**:
  $(\forall \chi)\mathcal{Q} \in \Gamma^*$ iff for each constant c, $\mathcal{Q}[c/\chi] \in \Gamma^*$

$\Rightarrow$ Assume $\mathfrak{M}^* \vDash (\forall \chi)\mathcal{Q}$. Show that $(\forall \chi)\mathcal{Q} \in \Gamma^*$

- Then every object satisfies $\mathcal{Q}$, so every substitution instance for every constant is true in $\mathfrak{M}^*$ (since each object is named by itself)
- These $\mathcal{Q}[c/\chi]$ are clubbin' by the IH, so they all belong to Γ^*. So then by Membership Lemma case (f), $(\forall \chi)\mathcal{Q} \in \Gamma^*$

$\Leftarrow$ Assume $(\forall \chi)\mathcal{Q} \in \Gamma^*$. Show that $\mathfrak{M}^* \vDash (\forall \chi)\mathcal{Q}$

- Practice this yourself!

Upshots of our Induction

- Having handled every case (in spirit), we conclude that every sentence of QL′ is clubbin’:
- For all QL′–sentences $\mathcal{P}$, $\mathfrak{M}^* \vDash \mathcal{P}$ iff $\mathcal{P} \in \Gamma^*$
- Hence, the QL′–model $\mathfrak{M}^*$ makes–true every sentence in Γ^*, showing that this set is satisfiable
- Hence, we have proven the **∃–C Maximal Consistency Lemma**: every QL′ set that is maximally–QND–consistent and ∃–complete is satisfiable in QL′
- It remains to prove the **Consistency Lemma**, i.e. that any QND–consistent QL–set (like our O.G. Γ) is satisfiable **in QL**!

From satisfiability of Γ^* to satisfiability of Γ

- We have shown that the maximally-QND-consistent and existentially complete Γ^* is satisfiable in QL$'$
- It remains to show that QND-consistent Γ is satisfiable **in QL**
- i.e. we need a QL-model $\mathfrak{M}$ s.t. $\mathfrak{M} \vDash \Gamma$
- **Hopes and dreams**: by construction $\Gamma \subset \Gamma^*$, so $\mathfrak{M}^* \vDash \Gamma$ in QL$'$. But how are we to get a QL-model for Γ from this??????
- **Salvation**: note that the model $\mathfrak{M}^*$ we constructed is *not only* a QL$'$ model for Γ^* *BUT ALSO* a QL-model for Γ!
- Since the language of QL is contained in QL$'$, $\mathfrak{M}^* := (D, I^*)$ maps all symbols of QL to objects in D
- If you like, you can define a QL-model $\mathfrak{M} := (D, I)$ s.t. I is the restriction of I^* to unprimed constants in QL. Then $\mathfrak{M} \vDash \Gamma$.
- $\square$ Q.E.D. MOST BLESSED STUDENTS!!! (i.e. *quod erat demonstrandum*)

Did we need to manually enforce ∃-completeness?

- In our condition (iii) for building up Γ^*, we manually enforced adding a substitution instance to our growing Γ_{k+1} whenever we add an existential sentence.
- Some have wondered: shouldn't condition (ii) take care of this? Substitution instances are QL′ sentences, so they arise at some k in our enumeration as well
- Really the issue is the following: are there maximally QND-consistent sets that are NOT existentially-complete? If so, then our condition (iii) is not idle
- So to show the necessity of our condition (iii) (or something like it), it suffices to construct a maximally QND-consistent set that has an existential sentence but no substitution instances for it.

A maximally QND-consistent but existentially INCOMPLETE set

- Let Γ_0 be the set $\{(\exists x){\sim}Fx\}$
- Its substitution instances have the form ${\sim}F[c/x]$, e.g. ${\sim}Fc$.
- Notice that the 'enemies' of these substitution instances always occur earlier in our enumeration, e.g. Fc occurs before ${\sim}Fc$, Fj'_{22} occurs before ${\sim}Fj'_{22}$ (the enemies always have one less symbol, so their index has two fewer digits)
- So imagine that we dropped condition (iii) and built up Γ^* using only conditions (i) and (ii).
- then Γ^* would contain $(\exists x){\sim}Fx$ and every instance of an 'enemy' substitution instance: $F\mathcal{c}$ for all constants $\mathcal{c}$
- Γ^* would NOT contain a single substitution instance of $(\exists x){\sim}Fx$ because every time we hit a ${\sim}F\mathcal{c}$ at its stage, Γ_k would already contain its enemy $F\mathcal{c}$, so that adding ${\sim}F\mathcal{c}$ would result in a QND-inconsistent set.

Some remaining concerns about this construction

- Intuitively, you might think that a set containing $(\exists x){\sim}Fx$ and all these enemies $F\mathcal{c}$ would be QND-inconsistent.
- But it is not! Note that from existential elimination, we cannot start our subproof with a constant occuring in a premise, and these 'enemies' would be premises. So there is no way to derive a contradiction
- Similarly, we can NOT go from an enemy to a contradictory universal $(\forall x)Fx$ because the constant can't occur in a premise
- Notice as well that the membership lemma would fail. We would have every instance of $F\mathcal{c}$ but Γ^* would NOT contain $(\forall x)Fx$ because this sentence is QND-inconsistent with $(\exists x){\sim}Fx$ (as an 8-line deduction shows)

14. Compactness of SL & QL

Soundness and Completeness

- Let Γ be any set of *sentences* of QL and Θ any sentence of QL.
- For our two natural deduction systems SND and QND, we have proven the following (where QND extends SND):
- **Soundness**: If $\Gamma \vdash_{QND} \Theta$, then $\Gamma \vDash \Theta$
  - QND derivations are 'safe' (they preserve truth)
  - (syntactic to semantic: i.e. we chose 'good' rules!)
- **Completeness**: If $\Gamma \vDash \Theta$, then $\Gamma \vdash_{QND} \Theta$
  - reasoning about arbitrary models is not needed to demonstrate validity: QND derivations suffice
  - (logical entailment is fully covered by our syntactic rules)

14. Compactness of SL & QL

a. Compactness of SL

Compactness of SL

- **Compactness of SL**: for any set Γ of SL-sentences (possibly infinite), Γ is satisfiable **if and only if** every finite subset $\Delta \subseteq \Gamma$ is satisfiable (i.e. for each Δ, there is a truth-value assignment that makes all sentences in Δ true).
- Relying on our valiant labors in proving the soundness and completeness of SND, we gain an elementary proof of compactness
- This proof is “impure” because it relies on syntactic notions, whereas the statement of compactness is purely semantic.

An “impure” proof of Compactness

- **Compactness of SL**: for any set Γ of SL–sentences, Γ is satisfiable **if and only if** every finite subset $\Delta \subseteq \Gamma$ is satisfiable

$\Rightarrow$ (trivial direction): Assume that Γ is satisfiable. Then there is a TVA that makes true every sentence in Γ.

- This TVA satisfies every finite subset $\Delta \subseteq \Gamma$.

$\Leftarrow$ (nontrivial direction): Assume that every finite subset $\Delta \subseteq \Gamma$ is satisfiable.

- Assume for *reductio* that Γ is unsatisfiable.
  Then there is no TVA that makes true every sentence in Γ.
- Hence, for any contradiction $\mathcal{C}$ (e.g. $P \,\&\, {\sim}P$), we have $\Gamma \vDash \mathcal{C}$

Impure proof: non-trivial direction continued

- (From above: Γ unsatisfiable $\Rightarrow \Gamma \models \mathcal{C}$, for contradiction $\mathcal{C}$)
- Hence, by completeness of SND, we can derive $\mathcal{C}$ from Γ: $\Gamma \vdash_{SND} \mathcal{C}$.
- Since derivations are finite, there exists a finite $\Delta \subseteq \Gamma$ such that $\Delta \vdash_{SND} \mathcal{C}$
- Then, by soundness of SND, $\Delta \models \mathcal{C}$. Since $\mathcal{C}$ is unsatisfiable, this means that Δ must be unsatisfiable as well.
- But that contradicts our starting assumption that every finite subset $\Delta \subseteq \Gamma$ is satisfiable.
- So Γ must be satisfiable (proving compactness)

What does compactness of SL tell us?

- *Question*: Are there any arguments of SL that have infinitely-many premises, where no premise is redundant?
- Assume that $\Gamma \vDash \mathcal{P}$. Then what can we say about $\Gamma \cup \{\sim\mathcal{P}\}$?

  $\Gamma \cup \{\sim\mathcal{P}\}$ is **unsatisfiable**!

  - So by one Contrapositive of Compactness, there exists a finite subset $\Delta \subset \Gamma \cup \{\sim\mathcal{P}\}$ that is unsatisfiable.
  - Easy to show that there is a finite $\Gamma_f \subset \Gamma$ s.t. $\Gamma_f \cup \{\sim\mathcal{P}\}$ is unsatisfiable as well. So $\Gamma_f \vDash \mathcal{P}$
- *Upshot*: every valid argument relies on finitely-many premises
- Contrast proof here with PS12 #4, which shows same result using completeness *and soundness*, relying on syntactic $\vdash_{SND}$
- Whereas our argument above proceeds entirely semantically, using compactness and semantic entailment $\vDash$
- If only we could prove compactness purely semantically?!

14. Compactness of SL & QL

b. A ‘Pure’ proof of SL compactness

A ‘Pure’ proof of the Compactness of SL

- Using a very similar idea to our construction of the maximally-SND-consistent set Γ^*, we can provide a purely semantic and yet still elementary proof of SL compactness
- Proof sketch: assuming that every finite subset of Γ is satisfiable, we will construct a superset $\Gamma^* \supset \Gamma$ for which it is easy to define a truth-value assignment that satisfies every sentence in Γ^*, and hence in Γ.
- As with our earlier completeness proof, Γ^* comes along with a membership lemma, which we use for our induction over SL.

Beginning the Proof

- $\Rightarrow$ (easy direction): assume that the (possibly infinite) set of SL-wffs Γ is satisfiable. Then there is a TVA that makes true every sentence in Γ, and this TVA satisfies every finite subset of Γ.
- $\Leftarrow$ (harder direction): Assume that every finite subset $\Delta \subset \Gamma$ is satisfiable. Show that Γ is satisfiable (nontrivial if Γ is infinite).
- ▶ Notice that it suffices to construct a superset Γ^* of Γ that is satisfiable. Then the TVA that makes true everything in Γ^* will make true everything in Γ.
- ▶ To proceed, we introduce an idea very similar to the notion of a maximally-consistent-in-SND set. But now using only *semantic* notions (so avoiding our proof system).

Maximally finitely satisfiable sets

- A set Γ^* of SL wffs is **maximally finitely satisfiable** (MFS) provided that:
  1.) Every finite subset of Γ^* is satisfiable (Γ^* is "**finitely satisfiable**")
  2.) For each SL wff $\mathcal{P}$, if $\Gamma^* \cup \{\mathcal{P}\}$ is FS, then $\mathcal{P} \in \Gamma^*$ ("*semantic Door*")
      Otherwise, adding any additional $\mathcal{P}$ to Γ^* breaks finite–satisfiability
      i.e. $\mathcal{P} \notin \Gamma^*$ iff $\Gamma^* \cup \{\mathcal{P}\}$ has an unsatisfiable finite subset

- Next we'll show that any MFS set is satisfiable (this mirrors our "maximal consistency lemma" from our completeness proof)
- To do this, we'll prove a membership lemma that facilitates an induction over SL!
- Finally, we'll show how to construct an MFS Γ^* from any finitely–satisfiable Γ
  (i.e. what we assume at the start of the nontrivial–direction)

Membership Lemma for MFS sets ("complete clubs")

- To induct on SL, we first show some constraints on Γ^* membership
- Basically, Γ^* has a a bouncer who enforces maximal finite satisfiability.
- **Membership Lemma** for club Γ^*: if $\mathcal{P}$ and $\mathcal{Q}$ are SL wffs, then:
  - a.) $\sim\mathcal{P} \in \Gamma^*$ if and only if $\mathcal{P} \notin \Gamma^*$
  - b.) $\mathcal{P} \,\&\, \mathcal{Q} \in \Gamma^*$ if and only if both $\mathcal{P} \in \Gamma^*$ and $\mathcal{Q} \in \Gamma^*$
  - c.) $\mathcal{P} \vee \mathcal{Q} \in \Gamma^*$ if and only if either $\mathcal{P} \in \Gamma^*$ or $\mathcal{Q} \in \Gamma^*$
  - d.) $\mathcal{P} \supset \mathcal{Q} \in \Gamma^*$ if and only if either $\mathcal{P} \notin \Gamma^*$ or $\mathcal{Q} \in \Gamma^*$
  - e.) $\mathcal{P} \equiv \mathcal{Q} \in \Gamma^*$ iff either (i) $\mathcal{P} \in \Gamma^*$ and $\mathcal{Q} \in \Gamma^*$ or (ii) $\mathcal{P} \notin \Gamma^*$ and $\mathcal{Q} \notin \Gamma^*$
- These syntactic constraints mirror truth-conditions, but we will now NOT rely on our proof system to prove this lemma
- (We built an analog of "the Door" into the definition of MFS sets)

Proof of Membership Lemma for MFS Sets

- **Case (a)**: $\sim\mathcal{P} \in \Gamma^*$ iff $\mathcal{P} \notin \Gamma^*$: use condition 2) (“semantic Door”) of MFS sets: $\mathcal{P} \notin \Gamma^*$ iff $\Gamma^* \cup \{\mathcal{P}\}$ has an unsatisfiable finite subset
- For the other cases, we rely on Case (a), the truth tables for the connectives, and the fact that Γ^* is finitely-satisfiable, i.e. every finite subset is satisfiable.
  (So we do lots of *reductio* proofs: assume that a membership case fails, apply Case (a), and then show this would result in an unsatisfiable finite subset—contradicting condition (1), i.e. that all finite subsets are satisfiable).
- So imagine we’ve proven the membership lemma!
- Then define a TVA $\mathcal{I}^*$ that makes true every atomic sentence in Γ^*;
  - show by induction that this TVA satisfies every sentence in Γ^*
  (just as in our proof of completeness of SND!)

Building an MFS Γ^* from a finitely-satisfiable Γ

- It remains to construct a maximally finitely-satisfiable superset Γ^* of a finitely-satisfiable Γ
- We first **enumerate** the SL wffs, so that every SL wff is associated with a unique positive integer $\{1, 2, 3, \ldots\}$
- Consider the first wff 'A' in our enumeration. If A can be added to Γ while preserving finite satisfiability, then let $\Gamma_1 := \Gamma \cup \{A\}$.
- Otherwise, let $\Gamma_1 := \Gamma$ (so that Γ_1 stays FS)
- Then, proceed to the second wff in our enumeration. If it can be added to Γ_1 without the new set breaking FS, let Γ_2 be the result. Otherwise, let $\Gamma_2 := \Gamma_1$
- Γ^* is the result of 'doing' this procedure for every SL wff
- More precisely, $\Gamma^* := \bigcup_{k=1}^{\infty} \Gamma_k$

Claim: Γ^* is maximally finitely satisfiable (MFS)

- At this point, it suffices to prove that Γ^* is MFS

1.) Clearly, Γ^* is finitely satisfiable. If it were not, then some $\Gamma_k \subset \Gamma^*$ would be finitely unsatisfiable, but that contradicts our construction conditions.

2.) Moreover, Γ^* is maximal: if there were a wff $\mathcal{Q}$ that could be added to Γ^* while preserving finite satisfiability, we would have added $\mathcal{Q}$ at its enumeration stage.

- So if $\mathcal{Q} \notin \Gamma^*$, it must be that $\Gamma^* \cup \{\mathcal{Q}\}$ is *not* finitely satisfiable.

- So we're done! Any finitely satisfiable Γ is a subset of an MFS Γ^*, which we've shown is satisfiable! So Γ is satisfiable!

14. Compactness of SL & QL

c. Compactness of First-order Languages

Compactness of QL

- **Compactness of QL**: for any set Γ of QL-sentences, Γ is satisfiable if and only if every finite subset $\Delta \subseteq \Gamma$ is satisfiable (i.e. $(\forall\Delta)\ \exists$ a QL-model $\mathfrak{M}_\Delta$ that makes true every sentence in Δ).
- *Mutatis mutandis*, we can provide an analogous impure proof, relying on the soundness and completeness of system QND

  And also a 'pure' proof, constructing a maximally finitely satisfiable and *existentially complete* superset Γ^*.
- To widen the interest of our results, let's generalize compactness to any first-order language $\mathcal{L}$

First-order Languages (FOLs)

- **First-order language $\mathcal{L}$**: a set of well-formed formulae specified by a recursion clause like the one we gave for QL, where the symbols of $\mathcal{L}$ include:
  - Variables: w, x, y, z (allowing subscripts $n \in \mathbb{N}$)
  - Operators: our five sentential connectives and two quantifiers
  - Punctuation: left and right parentheses
  - Names: a set of constants (allowing subscripts $n \in \mathbb{N}$)
  - Predicates: a non-empty set of capital letters (allowing subscripts), each with "an invisible label" giving its arity (e.g. 0-place, 1-place, 2-place, etc.)
  - a set of function symbols $f(c)$ (syntax: f maps terms to terms)
- Different FOLs differ in their names, predicates, and functions

$\mathcal{L}$-models and interpretations

- Let $\mathcal{L}$ be a first-order language, containing constants and k-place predicates (e.g. the language of QL)
  - recall that the atomic sentences of SL are 0th-place predicates
- An $\mathcal{L}$-model $\mathfrak{M} := (D, I)$ consists of
  1. A non-empty set D of objects, called the domain of $\mathfrak{M}$
  2. A map I (the *interpretation* of $\mathfrak{M}$), which maps the vocabulary of $\mathcal{L}$ to objects and ordered pairs from D as follows:
     - For each constant $c \in \mathfrak{L}$, $I(c)$ is an element of D, called the *referent* or denotation of c
     - For each k-place predicate P of $\mathfrak{L}$, $I(P)$ is a set of ordered k-tuples of objects in D, called the *extension* of P
     - I maps SL atomics to “true” or “false” (i.e. ‘1’ or ‘0’)

FOL with identity and functions

- With some minor modifications, we could extend our soundness and completeness proofs for QND to FOLs and deduction systems that include (1) a privileged identity predicate "$=$" and (2) functions that syntactically map terms to terms (interpreted as mapping the domain D to itself)
- Like our symbol "$\imath$", we add in some new symbol "α" that doesn't occur in our FOL, to give a countable infinity of unused constants
- To construct our maximally-syntactically-consistent, existentially complete superset Γ^*, we focus on equivalence classes of co-referential constants, since now some constants might name the same object in the domain (e.g. $c = d$)
- Using the axiom of choice, we could even handle FOLs that have *uncountably many* predicates or constants!

Compactness for a first-order language

- **Compactness of a FOL $\mathcal{L}$**: for any set Γ of $\mathcal{L}$-sentences (possibly infinite), Γ is satisfiable if and only if every finite subset $\Delta \subseteq \Gamma$ is satisfiable.

- We could prove this either by (i) using a soundness and completeness result for an $\mathcal{L}$-deduction system;
  (ii) generalizing our 'pure' proof for SL; or
  (iii) generalizing the topological proof of SL compactness (relying on results from topology, e.g. Tychonoff's theorem)

14. Compactness of SL & QL

d. The Löwenheim–Skolem theorems

Downwards!

- Terminology: we'll say that a model $\mathfrak{M}$ is *infinite* if its domain D is infinite in size. Likewise for saying that a model is finite, or countably infinite.
- **Downward Löwenheim–Skolem**: let Γ be a set of $\mathcal{L}$–sentences. If Γ is satisfiable in an infinite model, then it is satisfiable in a countably infinite model.
- *Gloss*: we can always descend from an infinite model to a countably infinite model
- Proof(s): (1) be impure and piggyback on completeness proof or (2) use compactness and satisfiability lemma for MFS sets

Down to be Impure

- **Converse consistency lemma**: if $\mathcal{L}$-set Γ is satisfiable, then Γ is syntactically-consistent (for a given deduction system $\mathcal{L}$ND that we've shown is sound)
- Proof: good exercise!!! Assume for *reductio* that Γ is syntactically-inconsistent and then apply soundness
- So since Γ is satisfiable, it is syntactically consistent.

  Then, appeal to our consistency lemma shown in the course of proving completeness: for any syntactically-consistent set, there is a maximally-consistent (and $\exists$-complete) set that is satisfiable, where we showed this by constructing a countably infinite model
- So Γ has a countably infinite model

Down with impurity: apply compactness

- *Pure proof of Downward LS*: assume that Γ is satisfied in an infinite model. Then it is satisfiable, and so by compactness theorem for FOL, Γ is finitely–satisfiable
- Modify our construction to form a maximally finitely–satisfiable and $\exists$–complete superset Γ^* of Γ
- Prove a satisfiability lemma: any such Γ^* is satisfiable, where we show this by constructing a countably infinite $\mathcal{L}^+$–model

  ($\mathcal{L}^+$ arises from $\mathcal{L}$ by adding a countable–infinity of new constants)
- Then, Γ has a countably infinite $\mathcal{L}$–model

Onwards and Upwards!

- **Upward Löwenheim–Skolem**: let Γ be a set of $\mathcal{L}$–sentences. If Γ is satisfiable in an infinite model $\mathfrak{M} := (D, I)$, then it is satisfiable in models of arbitrary size larger than $|D|$
- *Proof Sketch*: extend the set of constants $\mathcal{C}$ of $\mathcal{L}$ with an uncountable set $\mathcal{E}$ that contains $\mathcal{C}$.
  Extend the FOL $\mathcal{L}$ to $\mathcal{L}^+$ with $\mathcal{E}$ as its set of constants and with identity predicate $=$.
- Construct an $\mathcal{L}^+$–set Γ^+ by adjoining to Γ every sentence of the form $\sim c = d$ for every distinct $c, d \in \mathcal{E}$.
- Show that Γ^+ is finitely–satisfiable and hence by compactness satisfiable. Then note that any $\mathcal{L}^+$–model satisfying Γ^+ must have a domain as large $\mathcal{E}$. Restrict the interpretation function to construct an $\mathcal{L}$–model for Γ with domain $|D| = |\mathcal{E}|$

14. Compactness of SL & QL

e. Skolem’s ‘Paradox’

ZFC as a first-order language

- **Zermelo–Fraenkel set theory with choice** (ZFC): a FOL $\mathcal{ZFC}$ that has identity and a 2-place predicate for set-membership '$\in$', written between (rather than before) terms when forming atomic wffs
- In standard models, we interpret the objects as sets
- A list of axioms or axiom schemas, e.g.
  Null set axiom: $(\exists x)(\forall y)y \notin x$ (i.e. there is an empty set $\varnothing$)
  Axiom of Extensionality: $(\forall x)(\forall y)(\forall z)((z \in x \equiv z \in y) \supset x = y)$
  (i.e. two sets are identical iff they have the same members)
- **Axiom of Choice**: if x is a set whose members are non-empty sets and no two members of x share a member, then there is a set y that contains exactly one element of each set in x

Skolem's 'Paradox'

- If ZFC has any models, then it has a **countable model** (since by downward LS, an infinite model entails a countably infinite model. Any finite model is already countable—and can be extended to a countably infinite model as well)
- Yet, we can prove within ZFC that there are uncountable sets, e.g. the power set of $\mathbb{N}$ has cardinality of $\mathbb{R}$
- 'Paradox': how can a countably-infinite model make true the claim that there are uncountable sets?

Paradox Assuaged! (paradise regained?)

- Suppose that ZFC is satisfiable and so has a countable model $\mathfrak{M}$
- $\mathfrak{M}$ makes true all the axioms of ZFC and hence all the consequences of these axioms, including the claim U that says "the powerset of $\mathbb{N}$ is uncountable". Denote this set as '$2^{\mathbb{N}}$'
- U: there is an injection but no bijection from $\mathbb{N}$ to $2^{\mathbb{N}}$; $\mathfrak{M} \models U$
- Since $\mathfrak{M}$ is countable, the sets $\mathbb{N}$ and $2^{\mathbb{N}}$ in $\mathfrak{M}$ are definitely countable ($\mathfrak{M}$ has only countably many objects in its domain to serve as members of objects in that domain)
- So clearly, there IS a bijection between the sets that correspond to $\mathbb{N}$ and $2^{\mathbb{N}}$ in $\mathfrak{M}$ (we can prove this bijection in a metalanguage)
- BUT (resolution), this bijection is not itself an object in $\mathfrak{M}$. So $\mathfrak{M}$ itself represents $2^{\mathbb{N}}$ as uncountable

14. Compactness of SL & QL

f. Problems for finitism

Saying that there are finitely-many things

- As shown on PS13 problems #2, 5, and 6, we have some ISSUES when it comes to saying that there are finitely-many things in quantifier logic
- It seems like we definitely cannot accomplish this putatively possible task through *sentences*
- Is there any other way we might go about enforcing there being finitely-many things (e.g. if we think there probably are only finitely-many things and want a FOL to reflect that)?

Adding a finitely-many Quantifier

- If not through sentences, perhaps through operators, e.g. quantifiers!
- *Idea*: add a 'finitely-many' quantifier, Ǝ, to FOL
- Syntactically, we define Ǝ just like a quantifier: if $\mathcal{P}$ is a formula where x does not appear bound, then (Ǝx)$\mathcal{P}$ is a wff
- Semantically, we extend satisfiability semantics (oh boy—not that sh** again) so that (Ǝx)$\mathcal{P}$ is true in a model if and only if there are finitely-many $\mathcal{P}$-objects in the model's D, i.e. $|D|$ is finite
- **Question**: what would it take to modify our derivation system QND to make it sound and complete for quantifier logic with a finitely-many quantifier (QL-Ǝ)?
- **Answer**: no derivation system can be sound & complete for QL-Ǝ! - F***!!! INFINITE F***!!!

Finite Hopes & Finite Dreams: dashed upon ∞–many rocks

- Suppose for *reductio* that we had a sound and complete derivation system for QL–∄
- Then, we could prove that QL–∄ is compact (see slides 3–4)
- Yet, the entailment relation $\vDash_{QL-∄}$ for this logic is NOT compact:
- Consider the sentence $F := (∄x)x = x$, which says "there are finitely–many things that equal themselves." This is just a fancy way of saying that there are finitely–many things in the domain (since everything is identical to itself and nothing else).
- Then consider the set $X := \{F, L_1, L_2, \ldots\}$, containing F and each L_k for $k \in \mathbb{N}$, where L_k says "there are at least k–things"
- Set X is finitely–satisfiable, but it is not satisfiable (violating compactness). Any way of making true the infinitely–many L_n's requires an infinite model, which then can't make true sentence F

14. Compactness of SL & QL

g. A topological proof of SL compactness

What does “compactness” normally mean?

- **Topological space** (X, τ): a topology on a set X is a collection of **open sets** τ s.t. the following sets are open: (i) $\varnothing$ and X; (ii) arbitrary unions of open sets; (iii) finite intersections of open sets
- A set is closed in (X, τ) if its complement is open (NB: sets can be ‘clopen’, i.e. both open AND closed)
- Compactness in topology: a topological space is **compact** iff every open cover has a finite subcover
- Equivalently: every collection of closed subsets obeying the *finite intersection property* has non–empty intersection
- Finite intersection property (FIP): a set of subsets $\{F_\beta\}_{\beta \in B}$ of a topological space has the FIP if for every finite subset B_0 of our index set B, the intersection of all the sets F_β for $\beta \in B_0$ is non–empty, i.e. provided that $\bigcap_{\beta \in B} F_\beta$ is non–empty

Why call the logical property "compactness"?

- The compactness of SL is equivalent to the compactness of a particular topological space, namely a topology on the set of truth-value assignments (TVAs)
- Let $\mathcal{A}$ be the set of atomic wffs and let $\mathcal{E}$ be the set of TVAs
- for each atomic wff A, let U_A^0 be the set of TVAs that assign A false, and let U_A^1 be the set of TVAs that assign A true
- Endow the set $\mathcal{E}$ with a topology by stipulating that (i) for each atomic wff A, U_A^0 and U_A^1 are open and (ii) every non-empty open set arises as a union of these U^0s and U^1s
- **Claim**: the compactness of SL is equivalent to the compactness of this topological space $\mathcal{E}$
- Note that if we prove that 1) compactness of $\mathcal{E}$ entails compactness of SL and that 2) $\mathcal{E}$ is compact, then we will have proven compactness without detour through syntax!

Step 1: $\mathcal{E}$ compact entails SL is compact

- Assume that $(\mathcal{E}, \tau)$ is compact. Consider an arbitrary set Γ of SL sentences that is finitely satisfiable.

  NTS: Γ is satisfiable (the other direction is trivial)

- Consider an arbitrary wff $\mathcal{P}$. *Lemma*: the set $U_{\mathcal{P}} \subset \mathcal{E}$ of TVAs that make $\mathcal{P}$ true is open (proof: use disjunctive normal form and take a matching union of finite intersections of the U^0_As and U^1_Bs for atomics that compose $\mathcal{P}$!)

- So $U_{\sim\mathcal{P}}$ is also open. Since the complement of $U_{\mathcal{P}}$ is $U_{\sim\mathcal{P}}$, $U_{\mathcal{P}}$ is both closed and open

Step 1 continued: applying topological compactness

- So, for each wff $\mathcal{P}$ in Γ, the set of TVAs $U_{\mathcal{P}}$ that make $\mathcal{P}$ true is a closed subset of $\mathcal{E}$
- So to say that each finite subset Γ_0 of Γ is satisfiable is equivalent to saying that the family $\{U_{\mathcal{P}} : \mathcal{P} \in \Gamma\}$ is a family of closed subsets of $\mathcal{E}$ with the *finite intersection property* (i.e. for any finite subset of this family, the intersection of its members $U_{\mathcal{Q}}$ is non-empty)
- Since we are assuming that $\mathcal{E}$ is compact, the intersection of ALL members of this family $\{U_{\mathcal{P}} : \mathcal{P} \in \Gamma\}$ is non-empty
- i.e. this intersection must contain at least one TVA in $\mathcal{E}$
- Hence, there is a TVA that makes true all of the members of Γ

Step 2: show that $(\mathcal{E}, \tau)$ is compact

- We can think about $\mathcal{E}$ as equalling $2^{\mathcal{A}}$, i.e. the set of maps from the SL atomics $\mathcal{A}$ to the set $\{0, 1\}$
- Equip the set $\{0, 1\}$ with the discrete topology (i.e. every subset is open). Then the product topology on $2^{\mathcal{A}}$ equals the topology $(\mathcal{E}, \tau)$ defined earlier.
- Since there are countably many SL atomics, $2^{\mathcal{A}}$ is homeomorphic to the Cantor set (comprises ∞–binary sequences of 0s and 1s)
- Note that the Cantor set is compact, since it is a closed subset of a compact set (namely the closed unit interval [0, 1])
- If we allow $\mathcal{A}$ to have arbitrarily many SL atomics, then we could use Tychonoff's theorem (equivalent to the axiom of choice) to show that $2^{\mathcal{A}}$ is compact
- *Tychonoff*: a product of compact spaces is compact in the product topology

14. Final Review!

14. Final Review!

a. Checking Soundness for Alt. Rules

Alternative Natural Deduction Rules

- As we did with trees (system STD), we can consider whether modifying SND with a new rule preserves soundness

- Method for generating new cases: take a case in the book and add a negation symbol(s) somewhere;

  then figure out what a sound rule would give you.

Negated Conjunction Introduction

- Consider a system SND* just like SND except that we add the following rule:
- **Negated Conjunction Introduction**: from $\sim\mathcal{Q}$ derive $\sim(\mathcal{Q} \,\&\, \mathcal{R})$
- Does this rule preserve soundness? If so, extend our proof by adding a case to the induction (showing that the new line is righteous); If not, provide a concrete counterexample to soundness of SND*
- Strategy: first do a heuristic: do the earlier accessible sentences semantically entail the final sentence?
  - If yes, then the new rule preserves soundness (proceed to formally extend the proof!)
  - If no, then you should be able to construct a concrete counterexample to soundness (i.e. case where $\Gamma \vdash_{SND^*} P$ but $\Gamma \nvDash P$ for a concrete set of SL sentences Γ

Notation for Soundness Cases

- Γ_i stands for the set of assumptions that are open at the i–th line, i.e. these are the accessible premises/assumptions at line i. They are every premise/assumption (sentence sitting on a horizontal line) such that its scope line (vertical line) travels all the way down to line i, and line i is to the right of this vertical line.
- $\mathbf{P_i}$ stands for the sentence that is on the i–th line.
- $\Delta \subseteq \Gamma$ means that the set Δ is a subset of Γ.
- $\Gamma \cup \{\mathbf{Q}\}$ means that we have added the sentence $\mathbf{Q}$ to the set of sentences Γ (we have taken their union).

Induction Hypothesis and Key Fact

- *Induction hypothesis* for Soundness: assume that the soundness/righteousness property holds for all lines i less than the $k+1$–st line, i.e. if $i \leq k$ and if $\Gamma_i \vdash \mathbf{P_i}$, then $\Gamma_i \vDash \mathbf{P_i}$.
- In words: we are assuming that if we can derive a sentence $\mathbf{P_i}$ from a set of assumptions Γ_i, then those assumptions semantically entail that sentence.
- Lemma 6.3.2 (a.k.a. Useful Fact 1): if $\Gamma \vDash \mathbf{P}$ and Γ is a subset of a larger set Γ', then the larger set semantically entails the sentence $\mathbf{P}$ as well, i.e. $\Gamma' \vDash \mathbf{P}$.

Negated Conjunction Introduction

- **Negated Conjunction Introduction**: from $\sim\mathcal{Q}$ derive $\sim(\mathcal{Q} \,\&\, \mathcal{R})$
- Draw the deduction with the final sentence on line #k+1; label everything schematically so that you can refer to earlier line numbers and their open premise sets Γ_m
- Extend the proof of soundness by showing that a line generated by this rule is righteous, i.e. $\Gamma_{k+1} \vDash \mathcal{P}_{k+1}$
- Rely on the relevant subset relations between the various Γ premise sets
- Reason about relevant semantic entailment claims by using the truthtables for the connectives

Schematic Solution Steps (if you're totally lost)

1. Label the lines in your diagram with lowercase letters (e.g. j, ℓ, m, n, etc.) so that you can refer to them. Label the LAST LINE as $k+1$.
2. Reexpress the derivation diagram in terms of single turnstiles, i.e. if you have P on line j, then $\Gamma_j \vdash P$ (i.e. the set of open assumptions at line j provides a derivation for P).
3. Apply the induction hypothesis to any lines that are less than the $k+1$–th line. This lets you convert these single turnstiles into double turnstiles, e.g. $\Gamma_j \vDash P$, provided that $j < (k+1)$.
4. Relate the set of assumptions open at various lines (your Γ's) to the set of assumptions open at the last line, Γ_{k+1}. This will involve the subset relation $\subseteq$, e.g. $\Gamma_j \subseteq \Gamma_{k+1}$.
    - If the sentence P_j at line j is an additional open assumption that is not open at line $k+1$, then you need to tack this on, using the union operation: $\Gamma_j \subseteq (\Gamma_{k+1} \cup P_j)$.

Schematic Solution Steps continued

5. Apply useful fact 1 (i.e. lemma 6.3.2), using the relation(s) in the previous step. E.g., if you have $\Gamma_j \vDash P$ (from step 3) and $\Gamma_j \subseteq \Gamma_{k+1}$ (from step 4), then useful fact 1 entails that $\Gamma_{k+1} \vDash P$.

6. Next, consider an arbitrary truth value assignment that makes all of the sentences in Γ_{k+1} true. Use whatever double turnstiles are at your disposal to infer that some other sentence(s) is true.

6. Then, use a truth table to argue why the sentence on the last line (line $k+1$) must be true as well under this truth value assignment.

8. Pat yourself on the back (soundly)!

For additional guidance on Soundness, see...

- Section 6.3 of *The Logic Book* (reading for Week 12)
- pages 246–250 contain most of the cases for our system SND
- PS12 #1 handles negation elimination (case 10)
- §6.3 Exercises on page 250–251, problem #4 parts a thru d
- Think of your own cases by throwing in negation symbols, thinking about de Morgan's or other semantically equivalent sentences, etc.!

14. Final Review!

b. Applications of soundness & completeness

Applying soundness and/or completeness theorems

- PS12, problems #4–7 illustrate simple applications of the soundness and/or completeness theorems
- The final might contain problems of a similar flavor

A Key Fact to Remember, Understand, Retain

- If $\Gamma \cup \{\sim\mathcal{P}\}$ is unsatisfiable, what else can we say?
- Answer: $\Gamma \vDash \mathcal{P}$ (and vice-versa)
- If $\Gamma \vDash \sim\mathcal{Q}$, what else can we say?
- Answer: $\Gamma \cup \{\mathcal{Q}\}$ is unsatisfiable (and vice-versa)
- See p. 245 if you don't believe this; but should be able to give valid arguments for these claims verbally!

Practice w/ Applying Soundness & Completeness

To avoid ambiguity, let the sentences and sets of sentences be from QL, and let '$\vdash$' denote $\vdash_{QND}$

1. Prove or provide a counterexample to the following statement:
   If $\Gamma \vDash \mathcal{P}$ and $\Delta \vdash \mathcal{Q}$, then $\Gamma \cup \Delta \vdash \mathcal{P} \,\&\, \mathcal{Q}$
2. If $\Gamma \vdash (S \vee R)$ and $\Gamma \vdash \sim(S \vee R)$, prove or provide a counterexample that $\Gamma \vDash S$
3. If $\Gamma \cup \{\sim\mathbf{P}\}$ is unsatisfiable and $\Delta \vdash \mathbf{R}$, prove or provide a counterexample to $(\Gamma \cup \Delta) \vdash (\sim\mathbf{P} \equiv \mathbf{R})$.
4. Prove or give a counterexample to the following statement:
   If Γ is satisfiable, then $\{\sim S \mid S \in \Gamma\}$ is satisfiable.

Concept Review (if totally lost)

- Soundness theorem for SND: if you have a single turnstile (in SND), then you have a double turnstile. In words: if a set of assumptions gives you a derivation (in SND) for a sentence *S*, then those assumptions semantically entail that sentence *S*. In symbols: if $\Gamma \vdash_{SND} \mathbf{S}$, then $\Gamma \vDash \mathbf{S}$.
- Completeness theorem for SND: if you have a double turnstile, then you have a single turnstile (in SND). In words: if a set of assumptions semantically entails a sentence *S*, then those assumptions gives you a derivation (in SND) for that sentence *S*. In symbols: if $\Gamma \vDash \mathbf{S}$, then $\Gamma \vdash_{SND} \mathbf{S}$.
- Likewise for QL and QND

Solution Tips for Logically Complete Students

1. Use the soundness theorem to convert any single turnstiles you have (from system SND) into double turnstiles.
2. Convert claims about unsatisfiability into double turnstile relations
3. Use the completeness theorem to convert any double turnstiles you have into single turnstiles.
4. If you get stuck, write out the definitions of any key terms involved. These will guide you on your path to victory.
5. If you have to provide a counterexample, think about the simplest counterexample that gets the job done. Your counterexample must involve ACTUAL sentences; not metavariables
6. Pray for a stroke of insight! (Jk! Try reasoning backwards to figure out what you need!)

14. Final Review!

c. Alt. Cases of Membership Lemma

Alternative Cases of Membership Lemma

- In the completeness proofs, recall that the five SND membership lemma cases are motivated by truth-functional considerations

- We can prove variants of these cases, e.g. the following: modified version of case (e): $\sim \mathbf{P} \equiv \mathbf{Q} \in \Gamma^*$ if and only if either i) both $\mathbf{P} \notin \Gamma^*$ and $\mathbf{Q} \in \Gamma^*$ or ii) both $\mathbf{P} \in \Gamma^*$ or $\mathbf{Q} \notin \Gamma^*$.

- Alternately, one can be given an alternative SND rule (replacing one of our 11 sanctioned rules) from which to reprove a given case of the membership lemma (using the Door lemma)

Mega Reminder: TWO directions to show!

- Note that all of these cases have TWO directions, and you need to prove BOTH directions to complete the problem.

  First, you want to assume the thing on the left and derive the thing on the right (forward direction).

  Second, you want to assume the thing on the right and derive the thing on the left (backwards direction).
- Sometimes a case involves subcases, each of which can require its own non-trivial SND deduction (e.g. cases (c) and (d) for disjunction and conditional)
- Finally, remember that the membership lemma is purely syntactic! No mention of truth-value assignments here!

Membership Lemma (not that I'm a bouncer!)

- **Membership Lemma** for club Γ^*: if $\mathcal{P}$ and $\mathcal{Q}$ are SL wffs, then:
  - a.) $\sim\mathcal{P} \in \Gamma^*$ if and only if $\mathcal{P} \notin \Gamma^*$
  - b.) $\mathcal{P} \,\&\, \mathcal{Q} \in \Gamma^*$ if and only if both $\mathcal{P} \in \Gamma^*$ and $\mathcal{Q} \in \Gamma^*$
  - c.) $\mathcal{P} \vee \mathcal{Q} \in \Gamma^*$ if and only if either $\mathcal{P} \in \Gamma^*$ or $\mathcal{Q} \in \Gamma^*$
  - d.) $\mathcal{P} \supset \mathcal{Q} \in \Gamma^*$ if and only if either $\mathcal{P} \notin \Gamma^*$ or $\mathcal{Q} \in \Gamma^*$
  - e.) $\mathcal{P} \equiv \mathcal{Q} \in \Gamma^*$ iff either (i) $\mathcal{P} \in \Gamma^*$ and $\mathcal{Q} \in \Gamma^*$ or (ii) $\mathcal{P} \notin \Gamma^*$ and $\mathcal{Q} \notin \Gamma^*$
- Notice how these syntactic constraints mirror truth-conditions!

Two Examples of Membership Lemma

- Case (a) is very useful: $\sim\mathcal{P} \in \Gamma^*$ if and only if $\mathcal{P} \notin \Gamma^*$
- (modified version of case b):
  $\mathbf{P}\&\sim\mathbf{Q} \in \Gamma^*$ if and only if $\mathbf{P} \in \Gamma^*$ and $\mathbf{Q} \notin \Gamma^*$
- Additional practice problem (modified version of case c):
  prove that $\mathbf{P} \vee \sim\mathbf{Q} \in \Gamma^*$ if and only if either $\mathbf{P} \in \Gamma^*$ or $\mathbf{Q} \notin \Gamma^*$.
- Study case (d) for the conditional (bottom of p. 258)!
- Note that the 7-line derivation for case d) has a serious typo on line 2: the justification should be ":A / $\supset$ I", i.e. :AS for conditional intro.

Maximally Consistent-in-SND

- Γ^* is maximally-SND-consistent provided that both (i) Γ^* is consistent in SND (i.e. can't derive any contradictions) and (ii) if **P** is not in Γ^*, then $\Gamma^* \cup \{\mathbf{P}\}$ is inconsistent in SND.
- In other words: you can't derive a contradiction from assumptions in Γ^*. And if $\mathbf{P} \notin \Gamma^*$, then $\Gamma^* \cup \{\mathbf{P}\}$ lets you derive a contradictory pair (i.e. you can derive both **R** and $\sim\mathbf{R}$).
- Assuming that you are not asked to prove a variant of case (a), you can help yourself to this result. Hence, if a sentence $\mathbf{P} \notin \Gamma^*$, then case (a) lets you conclude that $\sim\mathbf{P} \in \Gamma^*$, and vice versa: if $\sim\mathbf{P} \in \Gamma^*$, then you can conclude that $\mathbf{P} \notin \Gamma^*$.

The Door Lemma

- Lemma 6.4.9 (a.k.a. 'The Door'): this lemma helps you show that a sentence S is a member of a maximally SND-consistent set Γ^*:
  - if you can derive S from a subset Γ of a maximally SND-consistent set Γ^*, then S is a member of Γ^*.
  - In symbols: if $\Gamma \vdash S$ and $\Gamma \subseteq \Gamma^*$, then $S \in \Gamma^*$. In particular, if $\Gamma^* \vdash S$, then $S \in \Gamma^*$.
  - Hence the strategy: if you are trying to show that $S \in \Gamma^*$, figure out how to derive S in SND from sentences you have assumed are in Γ^*. Then, apply The Door.

14. Final Review!

d. Translations in QL, with Identity

Some Structures to remember from SL

- **P only if Q**: $P \supset Q$ (order preserved) (equiv: $\sim Q \supset \sim P$)
- **Unless B, C** or **C unless B**: use OR: $B \vee C$
- **J just in case K**: $J \equiv K$
- Q if P; Q provided that P; Q given that P; if P, then Q: $P \supset Q$

Some Simple Examples not involving identity

Domain: all people;

Predicates: **Dx**: x went to Disneyland; **Kxy**: x knows y;

Constants/Names: j for John; m for Mary

- Schematize “Everyone who went to Disneyland knows someone who didn’t go there”:
  Answer: $(\forall x)(Dx \supset (\exists y)(Kxy \,\&\, {\sim}Dy))$
- “There is someone who knows both Mary and John but doesn’t know themself”:
  Answer: $(\exists x)(Kxm \,\&\, Kxj \,\&\, {\sim}Kxx)$
- “Everyone who knows John also knows Mary”:
  Answer: $(\forall x)(Kxj \supset Kxm)$

Singular "only"

- "**Only Greta** is a hero":

- Content: No–one other than Greta is a hero, **AND** Greta is a hero:

$(\forall x)(Hx \supset x{=}g)\ \&\ Hg$

$(\forall x)(Hx \equiv x{=}g)$

Schematizing ‘Neither’

- “Neither hero inspires”: this means that

  There are **exactly 2** heroes, and neither of them inspires:

$$(\exists x)(\exists y)\Big(\big((\sim x{=}y \;\&\; (Hx \;\&\; Hy)) \;\& \\ (\forall z)(Hz \supset (z = x \lor z = y))\big) \;\& \\ (\sim Ix \;\&\; \sim Iy)\Big)$$

At least *n*

- Remember: we interpret "three heros are inspiring" to mean "**at least** three heros are inspiring"
- At least 1 hero is inspiring:

$$(\exists x)(Hx \mathbin{\&} Ix)$$

- At least 2 heroes are inspiring:

$$(\exists x)(\exists y)({\sim}x{=}y \mathbin{\&} ((Hx \mathbin{\&} Ix) \mathbin{\&} (Hy \mathbin{\&} Iy)))$$

- At least 3 heroes are inspiring:

$$(\exists x)(\exists y)(\exists z)\Big(({\sim}x{=}y \mathbin{\&} ({\sim}y = z \mathbin{\&} {\sim}x = z)) \mathbin{\&} \\ ((Hx \mathbin{\&} Ix) \mathbin{\&} ((Hy \mathbin{\&} Iy) \mathbin{\&} (Hz \mathbin{\&} Iz)))\Big)$$

At least *n*

- Note: must state that **every pair** of variables is different, e.g.,

$$(\exists x_1)(\exists x_2)(\exists x_3)((\sim x_1 = x_2 \,\&\, \sim x_2 = x_3) \,\&$$
$$(Hx_1 \,\&\, (Hx_2 \,\&\, Hx_3)))$$

  only says “There are at least two heroes”!
  - Take extension of Hx to be: 1, 2
  - Then 1 can play role of x_1 and x_3, 2 role of x_2.
  - Both “$\sim 1 = 2$” and “$\sim 2 = 3$” are true.
- UD: People. Predicates: **Dx**: x went to Disneyland; **Kxy**: x knows y
- “There is somebody who went to Disneyland and knows at least two people who didn’t go there”
  Answer: $(\exists x)(Dx \,\&\, (\exists y)(\exists z)(\sim y = z \,\&\, Kxy \,\&\, Kxz \,\&\, \sim Dy \,\&\, \sim Dz))$

At most n

- There are **at most n** As $\Leftrightarrow$ There are **not at least $n+1$** As

$$(\exists^{\leq n} x)\, Ax \Leftrightarrow {\sim}(\exists^{\geq (n+1)} x)\, Ax$$

- For instance: There are at most two heroes:

$${\sim}(\exists x)(\exists y)(\exists z)((Hx \,\&\, (Hy \,\&\, Hz)) \,\&\, ({\sim}x = y \,\&\, ({\sim}x = z \,\&\, {\sim}y = z)))$$
$$(\forall x)(\forall y)(\forall z)((Hx \,\&\, (Hy \,\&\, Hz)) \supset (x = y \vee (x = z \vee y = z)))$$

- "At most one person who knows Mary doesn't know John"

  Answer: ${\sim}(\exists x)(\exists y)({\sim}x = y \,\&\, Kxm \,\&\, Kym \,\&\, {\sim}Kxj \,\&\, {\sim}Kyj)$

Definite descriptions

- Reminder: singular possessives like “Earth’s moon” can be interpreted like the definite description “the moon of Earth.” But plural possessives like “Mars’s moons” aren’t definite descriptions.
- Definite description: **the so-and-so**
- Russell’s analysis of definite description: to say

  “The *A* is B”

  is to say:
- There is something, which:
  - is *A*,
  - is the **only** *A* (i.e. the unique thing that is *A*),
  - is *B*.
- In QL:

$$(\exists x)(Ax \,\&\, (\forall y)(Ay \supset x{=}y) \,\&\, Bx)$$

Example: ‘The author of Waverley is blah’

- Schematize “The author of *Waverley* is Scottish”:
- Use the following symbolization key:
- Ax: x is an author; Wxz: x wrote z; Sx: x is Scottish; ℓ: *Waverley*

$$(\exists x)(Ax \mathbin{\&} Wx\ell \mathbin{\&} (\forall y)((Ay \mathbin{\&} Wy\ell) \supset x{=}y) \mathbin{\&} Sx)$$

Singular possessive (a definite description)

- Singular possessives form noun phrases, e.g., “Joe’s cape”
- They work like definite descriptions:
  Joe’s cape is the cape Joe owns. E.g.:
  - “Autumn wears Joe’s cape” symbolizes the same as:
    “Autumn wears the cape that Joe owns”:

$$
\begin{aligned}
(\exists x)\Big(\big((Ex \,\&\, Ojx)\,\&\, \\
&(\forall y)((Ey \,\&\, Ojy) \supset x{=}y)\big)\,\& \\
&Wax\Big)
\end{aligned}
$$

Singular vs. plural possessive

- Compare **plural** possessives: those are ‘$\forall$’s’:
  - “Autumn wears Joe’s cape**s**” symbolizes the same as:

    “Autumn wears every cape that Joe owns”:

$$(\forall x)\big((Ex \,\&\, Ojx) \supset Wax\big)$$

- So plural possessives are NOT definite descriptions.

14. Final Review!

e. Interpretations/Models for QL

Interpretations/Models for QL

- Study PS9, especially problems like 9, 10, 13, 15, 17, and 19
- Understand how to input this stuff into *Carnap*!
- Understand what it takes to make an existential statement true (at least one object in the domain must satisfy the statement)
- Understand what it takes to make a universal statement true (every object in the domain must satisfy the statement)
- Watch out for conditionals, which are trivially satisfied if the antecedent is false

Truth of sentences of QL

- Given an interpretation I ...
- An **atomic sentence** is true iff the referents of the constants are in the extension of the predicate:
  - Pa is true iff referent 'r' of a is in extension of P
  - Rab is true iff $\langle r, p\rangle$ is in extension of R (where r is referent of a, and p is referent of b)
- $\sim\mathcal{A}$ is true iff $\mathcal{A}$ is false
- $\mathcal{A} \vee \mathcal{B}$ is true iff at least one of $\mathcal{A}$, $\mathcal{B}$ is true
- $\mathcal{A} \,\&\, \mathcal{B}$ is true iff both $\mathcal{A}$, $\mathcal{B}$ are true
- $\mathcal{A} \supset \mathcal{B}$ is true iff $\mathcal{A}$ is false or $\mathcal{B}$ is true

Truth of quantified sentences

- $(\exists x)\,\mathcal{A}x$ is true iff $\mathcal{A}x$ is **satisfied** by **at least one** object in the domain
    - r satisfies $\mathcal{A}x$ in I iff $\mathcal{A}r$ is true in the interpretation

- $(\forall x)\,\mathcal{A}x$ is true iff $\mathcal{A}x$ is **satisfied** by **every** object in the domain

Truth of quantified sentences

- $(\exists x)\,(\mathcal{A}x \,\&\, \mathcal{B}x)$ is true iff some object satisfies '$\mathcal{A}x \,\&\, \mathcal{B}x$'
  - o satisfies '$\mathcal{A}x \,\&\, \mathcal{B}x$' iff it satisfies both $\mathcal{A}x$ and $\mathcal{B}x$
- $(\forall x)\,(\mathcal{A}x \supset \mathcal{B}x)$ is true iff **every** object satisfies '$\mathcal{A}x \supset \mathcal{B}x$'
  - o satisfies '$\mathcal{A}x \supset \mathcal{B}x$' iff either
    - o does not satisfy $\mathcal{A}x$ (vacuously true conditional)

      or
    - o does satisfy $\mathcal{B}x$

14. Final Review!

f. Derivations in QND

Quick Tips and a Practice Problem

- If you can do the derivations on PS10, you are probably in great shape!
- Focus in particular on the rules/syntax surrounding Existential Elimination, conditional introduction, and Universal Instantiation
- If you are going to do ∃E, it's typically best to start your proof with that and work within the ∃E subproof until you get what you need (which may not be what you want)
- Construct a deduction showing the following:
  $(\exists x)Qx, (\forall y)(Qy \supset Py) \vdash_{QND} (\exists z)Pz$
- Another practice problem!:
  $(\exists x)Gx \supset Fa \ \vdash_{QND} (\forall x)(Gx \supset Fa)$
- The other direction is MUCH trickier (but can be done in 10 lines)!
  $(\forall x)(Gx \supset Fa) \vdash_{QND} (\exists x)Gx \supset Fa$

Some General Advice (locate the MAIN quantifier)

- Make sure you have a firm grip on the four rules of QND, and the rules of SND (see PS6), and the rule sheet
- Make sure especially that you know how to correctly apply existential elimination (and check those three conditions) and universal introduction (and check those two conditions).
- There are no special conditions for universal elimination, and the one for existential introduction is technically enforced by our recursive defN of QL wffs
- Note that you CANNOT apply any of the SND rules *within* the scope of a quantifier! You must first eliminate the quantifiers to apply any SND rules to the stuff inside.
- In general, each rule applies only to the WHOLE sentence, not a part. So you CANNOT apply a rule to just part of a sentence.

Some more Specific Advice

- Make sure you understand how to build up a conditional by using conditional introduction!
  - Assume the conditional's antecedent, justified by :AS for $> I$
  - Derive the consequent in the scope of this assumption (possibly starting further subproofs to get to the consequent, like negation elimination)
  - Then discharge the assumption and write the conditional!
- You may then apply a quantifier rule to the conditional, e.g. Existential Introduction, to which you could then finish an existential elimination if you were in the scope of an EE subproof
- If you get stuck, try to work from the bottom up. Think about what you would first have to derive to build your ultimate goal.
- If you get stuck on a subgoal, assume the opposite of your subgoal to try using either negation introduction or negation elimination to keep going.