Homework Assignments for Logic-Based Natural Language Semantics (LBS WS22/23)

Michael Kohlhase, Jan Frederik Schaefer
FAU Erlangen-Nürnberg

2025-08-11

Contents

LBS WS 2024/25

Assignment1 – Epistemology & Syntax

Given: Oct. 31

Problem 1.1 (JTB-Definition of Knowledge) 10 pt

Consider the following scenario.

1. Smith and Jones have applied for a certain job.
2. The president of the company assured Smith that Jones would, in the end, be selected for the job.
3. Smith had counted ten coins in Jones' pocket ten minutes ago.
4. Thus Smith concludes that the man who gets the job has ten coins in his pocket.
5. In the end however, unknown to Smith, he himself, not Jones is selected for the job.
6. Also, unknown to Smith, he himself has ten coins in his pocket.

According to the JTB-definition of knowledge, does Smith *know* that his conclusion (statement **??** above) is true? Explain your answer.

Solution: Smith does not know sentence **??**. It is true in the scenario, but not justified, because Smith does not know sentences **??** and **??**, which could have justified sentence **??**.

Problem 1.2 (Most Certain Principle) 10 pt

State Cresswell's Most Certain Principle and discuss why it is important in the LBS context (1–3 sentences).

Solution: Cresswell's Most Certain Principle states that if you have two sentences, A and B and one of them is true while the other one is false, then they must mean different things. We can use it to see if our meaning theory is sufficiently fine-grained.

(maybe also:) Together with the congruence principle, it gives us a synonymy test.

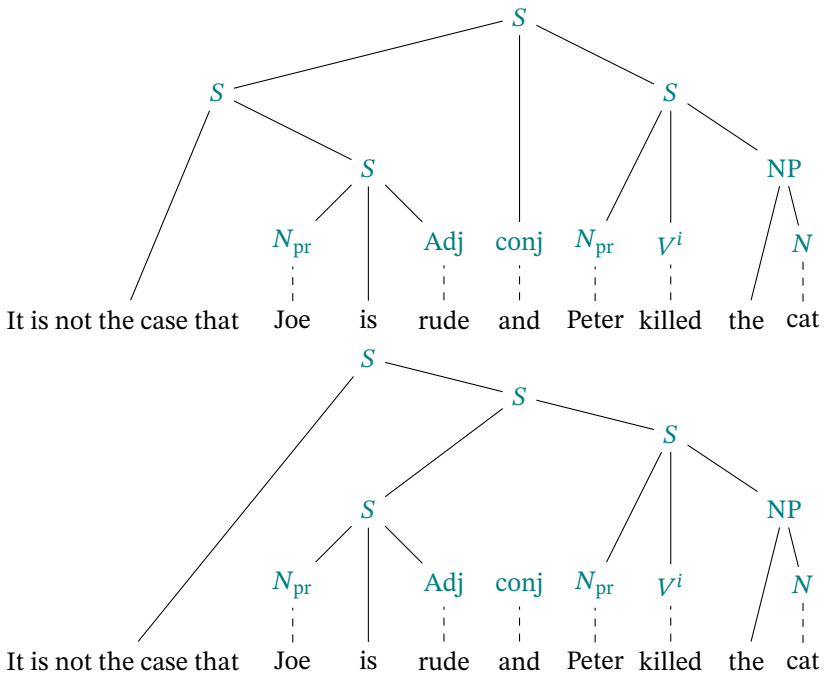
Problem 1.3 (Syntax Tree) 10 pt

Consider the following production rules:

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

Draw two syntax trees for the sentence "*It is not the case that Joe is rude and Peter killed the cat.*" according to the structural rules above; assume whatever lexical rules you need.

Solution: The lexical rules are represented by the dashed edges in the parse trees below.



Problem 1.4 (Grammar from Data) 10 pt

Write a context-free grammar that accepts the following sentences:

1. "*There is a pit in sector e.*"
2. "*There is no pit in sector g.*"
3. "*There is an echo in sector f and there is no echo in sector a.*"

4. "*There are echoes in sectors f and g and there are no pits in sector c.*"
5. "*There is a pit in sector a or there are pits in sectors b and c.*"
6. "*There are no pits in sectors c, a, e and d and there is an echo in sector g.*"

and reject these:

1. "*There is an pit in sector b.*"
2. "*There is an echo in sector b and.*"
3. "*There is a pit in sectors a.*"
4. "*There are pits in sectors d.*"
5. "*There are pits in sectors a, b and f, c.*"
6. "*There are pits in sectors g, a.*"

Explain your grammar design.

Solution:

```
<S> ::= "There " <Smain> "."
<Smain> ::= <Sconj> | <Ssingular> | <Splural>
<Sconj> ::= <Smain> " and there " <Smain>
<Ssingular> ::= <Psing> " in sector " <Sector>
<Psing> ::= "is a pit" | "is no pit" | "is an echo" | "is no echo"
<Sector> ::= "a" | "b" | "c" | "d" | "e" | "f" | "g"
<Splural> ::= <Pplural> " in sectors " <SecList> " and " <Sector>
<Pplural> ::= "are pits" | "are no pits" | "are echoes" | "are no echoes"
<SecList> ::= <Sector> | <Sector> ", " <SecList>
```

The toplevel category is `<S>`. `<Smain>` describes a sentence without the leading "*there*". `<Ssingular>` is for singular `<Smain>`s (e.g. "*is no echo in sector b*"), while `<Splural>` is for plural ones.

LBS WS 2024/25

Assignment2 – Fragment 1

Problem 2.1

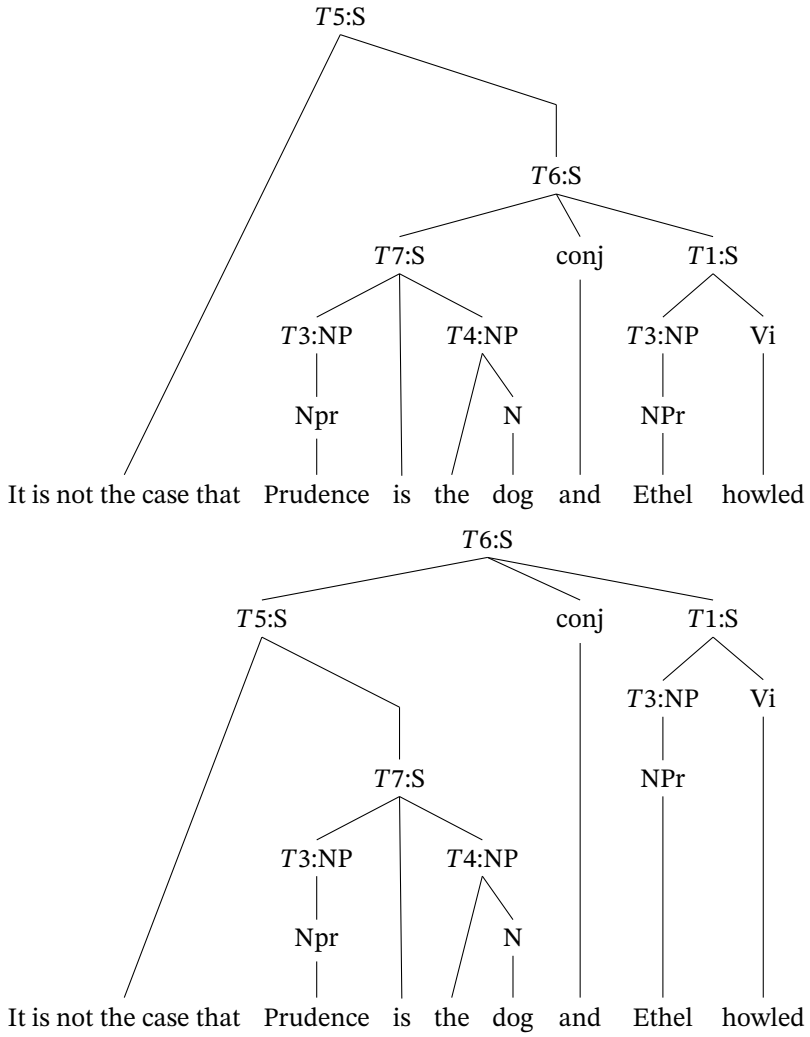
Consider the following sentence:

"*It is not the case that Prudence is the dog and Ethel howled.*"

1. How would the sentence be parsed according to the grammar of Fragment 1? 0 pt

---

Solution: The sentence is syntactically ambiguous. Here are the two possible parses (the nodes are already marked with the relevant translation rules):



2. Apply the translation rules/semantics construction from Fragment 1 to the sentence. 0 pt

Solution: As we have two possible parses, we will have two possible translations. Here they are the results:

- $\neg(Prudence' = thedog' \wedge howled'(Ethel'))$
- $\neg(Prudence' = thedog') \wedge howled'(Ethel')$

Problem 2.2

Extend a) the grammar and b) the translation rules of Fragment 1 to include more sentences. Concretely, with your extension the grammar of fragment 1 should accept sentences like

"*Prudence isn't the dog.*" "*If Ethel isn't crazy, then Prudence is the dog.*"

Solution: New production rules:

$S7:\ S \to \text{NP}, \text{isn't}, \text{NP}$
$S8:\ S \to \text{NP}, \text{isn't}, \text{Adj}$
$S9:\ S \to \text{If}, S, \text{then}, S$

New translation rules:

$T7:\ [X_{\text{NP}}, \text{isn't}, Y_{\text{NP}}]_S \rightsquigarrow (\neg X' = Y')$
$T8:\ [X_{\text{NP}}, \text{isn't}, Y_{\text{Adj}}]_S \rightsquigarrow (\neg Y'(X'))$
$T9:\ [\text{If}, X_S, \text{then}, Y_S]_S \rightsquigarrow (X' \Rightarrow Y')$

Problem 2.3 (PLNQ Semantics)

Consider the first-order signature

$$\Sigma_0^f = \{o\} \tag{1}$$
$$\Sigma_1^f = \{s\} \tag{2}$$
$$\Sigma_1^p = \{p, z\} \tag{3}$$

and the following PLNQ formula

$$\varphi := p(s(o)) \wedge \neg z(o)$$

1. Evaluate φ using the model $\langle \mathcal{D}, \mathcal{I} \rangle$ with 0 pt

$$\mathcal{D} = \{■, ♦, ★\} \tag{4}$$
$$\mathcal{I}(o) = ♦ \tag{5}$$
$$\mathcal{I}(s) = \{■ \mapsto ♦, ♦ \mapsto ■, ★ \mapsto ★\} \tag{6}$$
$$\mathcal{I}(p) = \{■, ★\} \tag{7}$$
$$\mathcal{I}(z) = \{■, ★\} \tag{8}$$

Solution: The expression evaluates to T. Explanation:

1. $\mathcal{I}(p(s(o)) \wedge \neg z(o)) = \mathsf{T}$, iff
2. $\mathcal{I}(p(s(o)))$ and $\mathcal{I}(\neg z(o))$ are both T
3. $\mathcal{I}(p(s(o))) = \mathsf{T}$ because
   (a) $\mathcal{I}(p(s(o))) = \mathsf{T}$, iff
   (b) $\mathcal{I}((s)(o)) \in \mathcal{I}(p)$, iff
   (c) $\mathcal{I}(s)(\mathcal{I}(o)) \in \{■, ★\}$, iff
   (d) $\mathcal{I}(s)(♦) \in \{■, ★\}$, iff
   (e) $■ \in \{■, ★\}$
4. $\mathcal{I}(\neg z(o)) \neq \mathsf{T}$ because
   (a) $\mathcal{I}(\neg z(o)) = \mathsf{T}$, iff
   (b) $\mathcal{I}(z(o)) = \mathsf{F}$, iff
   (c) $\mathcal{I}(o) \in \mathcal{I}(z)$, iff
   (d) $♦ \in \{■, ★\}$, which is false.

---

2. Evaluate φ using the model with: 0 pt

$$\mathcal{D} = \{0, 1, 2, 3, ...\} \tag{9}$$
$$\mathcal{I}(o) = 0 \tag{10}$$
$$\mathcal{I}(s) = \{0 \mapsto 1, 1 \mapsto 2, 2 \mapsto 3, ...\} \tag{11}$$
$$\mathcal{I}(p) = \{1, 2, 3, ...\} \tag{12}$$
$$\mathcal{I}(z) = \{0\} \tag{13}$$

In other words, the domain is the natural numbers, o is interpreted as 0, s is interpreted as the successor function, p is interpreted as the set of all positive integers, and z is interpreted as the set containing only 0.

Solution: Sketch: $\mathcal{I}(p(s(o))) = \mathsf{T}$ because $\mathcal{I}(s(o)) = 1 \in \{1, 2, 3, ...\}$, and $\mathcal{I}(\neg z(o)) = \mathsf{F}$ because $0 \in \{0\}$. Therefore, $\mathcal{I}(\varphi) = \mathcal{I}(\wedge)(\mathsf{T}, \mathsf{F}) = \mathsf{F}$.

LBS WS 2024/25

Assignment3 – Fragment 2

Problem 3.1 (Recap: Model generation with propositional tableaux)
Use the propositional tableau calculus to generate a models for the propositional formula.

$$(A \vee B) \wedge \neg(B \wedge C) \wedge (\neg C \vee \neg A)$$

Hint: The formula above has conjunction with three conjuncts. For your tableau you may want to add parentheses so that the tableau rules for the (binary) connectives apply more easily.

Solution: We construct a saturated tableau:

$$\begin{array}{c}
((A \vee B) \wedge \neg(B \wedge C) \wedge (\neg C \vee \neg A))^{\mathsf{T}} \\
(A \vee B)^{\mathsf{T}} \\
(\neg(B \wedge C) \wedge (\neg C \vee \neg A))^{\mathsf{T}} \\
\neg(B \wedge C)^{\mathsf{T}} \\
(\neg C \vee \neg A)^{\mathsf{T}} \\
(B \wedge C)^{\mathsf{F}} \\
\begin{array}{cccc}
A^{\mathsf{T}} & & B^{\mathsf{T}} & \\
\neg C^{\mathsf{T}} & \neg A^{\mathsf{T}} & \neg C^{\mathsf{T}} & \neg A^{\mathsf{T}} \\
C^{\mathsf{F}} & A^{\mathsf{F}} & C^{\mathsf{F}} & A^{\mathsf{F}} \\
B^{\mathsf{F}} \mid C^{\mathsf{F}} & B^{\mathsf{F}} \mid C^{\mathsf{F}} & B^{\mathsf{F}} \mid C^{\mathsf{F}} & B^{\mathsf{F}} \mid C^{\mathsf{F}} \\
\square \mid \square & \bot \mid \bot & \bot \mid \square & \bot \mid \square
\end{array}
\end{array}$$

So we have four closed branches (they end in $\bot$), and four open ones (decorated by $\square$), these correspond to propositional models.

Problem 3.2 (Tableau Machine)
Consider the sentence
"*The dog chased the cat. It climbed up the tree.*"

1. Construct a model generation tableau to represent the following discourse, incorporating only information contained in the sentences. 4 pt

Hint: You can treat "*climbed up*" as a complex transitive verb with translation climb.

Make sure that you use a suitable (compositional) representation of definite descriptions.

Solution:

$$\boxed{chase(\iota\ \text{dog}, \iota\ \text{cat})}$$
$$\boxed{climb(X, \iota\ \text{tree})}$$
$$climb(\iota\ \text{dog}, \iota\ \text{tree})^{\mathsf{T}} \mid climb(\iota\ \text{cat}, \iota\ \text{tree})^{\mathsf{T}} \mid climb(\iota\ \text{tree}, \iota\ \text{tree})^{\mathsf{T}}$$

2. How many possible readings are predicted? 1 pt

Solution: Three (no branch closes)

3. Now modify the tableau by including a representation of the world knowledge that the dog does not climb up anything. 3 pt

Solution:

$$\neg climb(\iota\ \text{dog}, Y)^{\mathsf{T}}$$
$$\boxed{chase(\iota\ \text{dog}, \iota\ \text{cat})}$$
$$\boxed{climb(X, \iota\ \text{tree})}$$

$climb(\iota\ \text{dog}, \iota\ \text{tree})^{\mathsf{T}}$ $\neg climb(\iota\ \text{dog}, \iota\ \text{tree})^{\mathsf{T}}$ $climb(\iota\ \text{dog}, \iota\ \text{tree})^{\mathsf{F}}$ $\bot$	$climb(\iota\ \text{cat}, \iota\ \text{tree})^{\mathsf{T}}$	$climb(\iota\ \text{tree}, \iota\ \text{tree})^{\mathsf{T}}$

4. How would the tableau machine for natural language understanding deal with the situation where we have multiple open branches? 2 pt

Solution: It would choose a preferred branch and focus on the reading/model it represents by adding new information to that branch and saturating the subtableau – backtracking on the branch choice if the new subtableau closes.

5. Finally, what do we have to add as world knowledge (based on the concept that trees are plants; do not just state that "trees – like dogs above – do not climb anything) to make sure that we only get one reading as intuitively expected. 2 pt
   You do not have to draw the tableau, just state the world knowledge.

Solution:

1. Trees are plants.
2. Plants cannot move.
3. If something cannot move, it cannot climb.

Problem 3.3 (Problems with fragment $\mathcal{F}_2$)

1. Consider the following discourse 0 pt

   "*Peter lives in Edinburgh. He has a dog John.*"

   Let us assume the following translation to logic

$$livein(peter, edinburgh)$$

$$have(X, john) \wedge dog(john)$$

   and the world knowledge

$$(have(A,B) \wedge dog(B) \Rightarrow human(A))^{\mathsf{T}}$$

$$(city(C) \Rightarrow \neg human(C))^{\mathsf{T}}$$

$$(dog(D) \Rightarrow \neg human(D))^{\mathsf{T}}$$

$$city(edinburgh)^{\mathsf{T}}$$

   Construct a model generation tableau.

Solution:

$(have(A,B) \wedge dog(B) \Rightarrow human(A))^{\mathsf{T}}$
$(city(C) \Rightarrow \neg human(C))^{\mathsf{T}}$
$(dog(D) \Rightarrow \neg human(D))^{\mathsf{T}}$
$city(edinburgh)^{\mathsf{T}}$

$\boxed{livein(peter, edinburgh)}$
$\boxed{have(X, john) \wedge dog(john)}$

$(dog(john) \Rightarrow \neg human(john))^{\mathsf{T}}$
$(city(edinburgh) \Rightarrow \neg human(edinburgh))^{\mathsf{T}}$
$\neg human(edinburgh)^{\mathsf{T}}$
$human(edinburgh)^{\mathsf{F}}$

$(have(peter, john) \wedge dog(john))^{\mathsf{T}}$	$(have(edinburgh, john) \wedge dog(john))^{\mathsf{T}}$	$(have(john, john) \wedge dog(jo$
$human(peter)^{\mathsf{T}}$	$human(edinburgh)^{\mathsf{T}}$	$human(john)^{\mathsf{T}}$
$dog(john)^{\mathsf{T}}$	$\bot$	$dog(john)^{\mathsf{T}}$
$\neg human(john)^{\mathsf{T}}$		$\neg human(john)^{\mathsf{T}}$
$human(john)^{\mathsf{F}}$		$human(john)^{\mathsf{F}}$
		$\bot$

---

2. Now let us assume that we do not know the name of the dog: 0 pt

"*Peter lives in Edinburgh. He has a dog.*"

We can introduce a variable Y for the dog instead:

$$livein(peter, edinburgh)$$

$$have(X, Y) \wedge dog(Y)$$

Why does the model generation not work in this case? How could it be fixed?

Solution: Applying $\mathcal{T}^p_V Ana$ to $\boxed{have(X,Y) \wedge dog(Y)}$ would yield the following new branches:

1. $(have(peter, peter) \wedge dog(peter))^{\mathsf{T}}$
2. $(have(peter, edinburgh) \wedge dog(edinburgh))^{\mathsf{T}}$
3. $(have(edinburgh, peter) \wedge dog(peter))^{\mathsf{T}}$
4. $(have(edinburgh, edinburgh) \wedge dog(edinburgh))^{\mathsf{F}}$

Only the second branch would not close.
Essentially, the problem is that $\mathcal{T}^p_V Ana$ only uses existing constants, but in this case there is no appropriate constant for Y. To fix this, we could extend $\mathcal{T}^p_V Ana$ to also allow the introduction of new constants.
Then it would generate the following branches:

1. $(have(peter, peter) \wedge dog(peter))^{\mathsf{T}}$ (closes)
2. $(have(peter, edinburgh) \wedge dog(edinburgh))^{\mathsf{T}}$ (closes)

3. $(have(peter, c_1) \wedge dog(c_1))^{\mathsf{T}}$ (the reading we want)
4. $(have(edinburgh, peter) \wedge dog(peter))^{\mathsf{T}}$ (closes)
5. $(have(edinburgh, edinburgh) \wedge dog(edinburgh))^{\mathsf{T}}$ (closes)
6. $(have(edinburgh, c_2) \wedge dog(c_2))^{\mathsf{T}}$ (closes)
7. $(have(c_3, peter) \wedge dog(peter))^{\mathsf{T}}$ (does not close)
8. $(have(c_4, edinburgh) \wedge dog(edinburgh))^{\mathsf{T}}$ (closes)
9. $(have(c_5, c_6) \wedge dog(c_6))^{\mathsf{T}}$ (does not close)

The three readings that do not close are, arguably, reasonable (to get a constant for "*he*", imagine someone pointing at a bystander and saying "***he** has a dog*"). We could use a heuristic to determine which reading is more likely to be correct.

---

3. Let us consider the following piece of (hypothetical) world knowledge: 0 pt

   "*Every human who lives in Edinburgh has a dog.*"

   We could represent it as

   $$human(H) \wedge live(H, edinburgh) \Rightarrow have(H, D) \wedge dog(D)$$

   Why is this problematic? What would go wrong if you add this world knowledge to the first subproblem of this problem?

Solution: All branches would close. Observe what happens with the branch of the intended reading where Peter has the dog:

$$(have(peter, john) \wedge dog(john))^{\mathsf{T}}$$

$$human(peter)^{\mathsf{T}}$$

$$human(peter) \wedge live(peter \wedge edinburgh) \Rightarrow have(peter, D) \wedge dog(D)$$

$$human(peter) \wedge live(peter, edinburgh) \Rightarrow have(peter, edinburgh) \wedge dog(edinburgh)$$

Now we will get a contradiction, as we can prove that Peter is a dog (which implies that Peter is not a human). The problem is that we could instantiate D with anything we want according to $\mathcal{T}^{p}_{\forall} WK$, including $edinburgh$.

To mark that D is not universally quantified, but rather existentially quantified, we need a more powerful logic. First-order logic is the natural candidate.

LBS WS 2024/25

Assignment4 – Equality and Tableaux

Problem 4.1 (Commutativity of equality)
Prove the following formula in $\mathcal{T}^{=}_{\mathrm{NQ}}$:

$$a = b \Rightarrow b = a$$

Solution:

0.	$(a = b \Rightarrow b = a)^{\mathsf{F}}$	start
1.	$a = b^{\mathsf{T}}$	from 0
2.	$b = a^{\mathsf{F}}$	from 0
4.	$a = a^{\mathsf{F}}$	from 1 and 2
5.	$a = a^{\mathsf{T}}$	-
6.	$\bot$	from 4 and 5

Problem 4.2 (Tableaux and equality)
Prove the following formula in $\mathcal{T}^{=}_{\mathrm{NQ}}$:

$$a = b \Rightarrow (b = c \Rightarrow c = a)$$

Solution:

0.	$(a = b \Rightarrow (b = c \Rightarrow c = a))^{\mathsf{F}}$	start
1.	$a = b^{\mathsf{T}}$	from 0
2.	$(b = c \Rightarrow c = a)^{\mathsf{F}}$	from 0
3.	$b = c^{\mathsf{T}}$	from 2
4.	$c = a^{\mathsf{F}}$	from 2
5.	$c = b^{\mathsf{F}}$	from 1 and 4
6.	$c = c^{\mathsf{F}}$	from 3 and 5
7.	$c = c^{\mathsf{T}}$	-
8.	$\bot$	from 6 and 7

Assignment5 – Lambdas and Fragment 5

Problem 5.1 (Church Booleans)

Boolean logic can be encoded in the lambda calculus. Concretely, we will denote truth values as functions:

$$false := \lambda x.\lambda y.y$$
$$true := \lambda x.\lambda y.x$$

1. Let 0 pt

$$or := \lambda a.\lambda b.a\ a\ b$$

Compute $or\ false\ true$, i.e. beta reduce

$$(\lambda a.\lambda b.a\ a\ b)\ (\lambda x.\lambda y.y)\ (\lambda x.\lambda y.x)$$

Hint: The result should be true, i.e. a function of the form $\lambda x.\lambda y.x$.

Solution:

$$\begin{aligned}
&(\lambda a.\lambda b.a\ a\ b)\ (\lambda x.\lambda y.y)\ (\lambda x.\lambda y.x)\\
\rightsquigarrow_\beta &(\lambda b.(\lambda x.\lambda y.y)\ (\lambda x.\lambda y.y)\ b)\ (\lambda x.\lambda y.x)\\
\rightsquigarrow_\beta &(\lambda x.\lambda y.y)\ (\lambda x.\lambda y.y)\ (\lambda x.\lambda y.x)\\
\rightsquigarrow_\beta &(\lambda y.y)\ (\lambda x.\lambda y.x)\\
\rightsquigarrow_\beta &\lambda x.\lambda y.x
\end{aligned}$$

---

2. What should be the function for the *not* operator? 0 pt

Hint: It helps to have a feeling for $true$ and $false$. They are both functions that accept two arguments, x and y. $true$ returns the first argument, x, while $false$ returns the second argument, y.
The function *or* maps two booleans, a and b to the expression $a\ a\ b$. The "top level" function is a, and it gets the arguments a and b. If a is true, the

expression will evaluate to the first argument, a. In other words, if a is true, the expression will evaluate to true. If a is false, the expression will evaluate to the second argument, b. So if both a and b are false, it will evaluate to false, but if b is true, it will evaluate to true.
We could also have defined *or* as

$$or = \lambda a.\lambda b.a\ true\ b$$

Solution: Option 1:

$$not := \lambda a.a\ false\ true$$

if a is true, $a\ false\ true$ will evaluate to the first argument, $true$, otherwise it will evaluate to the second argument, $false$.
Option 2:

$$not := \lambda a.\lambda x'.\lambda y'.a\ y'\ x'$$

here the intuition is that we "make a new truth function from scratch" with arguments x' and y' (corresponding to x and y in $true$ and $false$). However, we swap the arguments.

---

3. What should be the function for the *nor* operator ($X\ nor\ Y$ is true iff neither X nor Y are true)? 0 pt

---

Solution:

$$nor := \lambda a.\lambda b.a\ false\ (not\ b)$$

Explanation: If a is true, it will evaluate to the first expression, $false$. Otherwise, it will evaluate to $not\ b$.
Alternative solution:

$$nor := \lambda a.\lambda b.not\ (or\ a\ b)$$

Problem 5.2 (Church Booleans)
Given the following lambda expressions

$$p\ (f\ x\ x)$$
$$p\ (h\ (p\ y))$$
$$p\ ((\lambda k.g\ k)x\ )$$
$$p\ (m\ (\lambda l.p\ l))$$

and the types

$$p : \iota \to o$$
$$x : \iota$$

What types must the other symbols have for the expressions to type-check?

Solution:

$$f : \iota \to \iota \to \iota$$
$$y : \iota$$
$$h : o \to \iota$$
$$g : \iota \to \iota$$
$$m : (\iota \to o) \to \iota$$

and for the bound variables

$$k : \iota$$
$$l : \iota$$

Problem 5.3

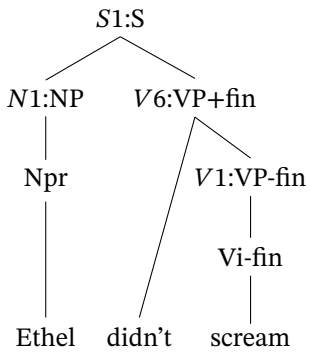
Consider the following sentence:

"*Ethel didn't scream.*"

1. How would the sentence be parsed according to the grammar of Fragment 3? 0 pt

---

Solution:



---

2. Apply the translation rules/semantics construction from Fragment 3 to the sentence. 0 pt

---

Hint: The translation rules are not clearly specified in the slides. Try to understand the ideas behind fragment 3 to fill in the gaps.

Solution: On the top-level, we apply the translation of the verb phrase to the translation of the noun phrase. The translation of the noun phrase is $ethel'$. The translation of the verb phrase is the translation of "*didn't*", which is $\lambda P.\lambda X.\neg\ P\ X$, applied $scream'$. So we get

$$((\lambda P.\lambda X.\neg\ P\ X)\ scream')\ ethel'$$

which reduces to

$$\neg\ scream'\ ethel'$$

LBS WS 2024/25

Assignment6 – Noun Phrases in Fragment 4

Problem 6.1

1. In fragment 4, noun phrases are translated to expressions of type $(\iota \to o) \to o$. What was the motivation for this? What can we do now that was not possible before? 0 pt
2. Names (N_{pr}) are still of type ι. How would we translate the *noun phrase* "*Ethel*"? (the result should be an expression of type $(\iota \to o) \to o$) 0 pt
3. Parse the sentence "*every dog saw the red cat*" and apply the translation rules according to fragment 4. The translation is not fully specified in the slides, but if you understood the previous fragments and the handling of noun phrases, you should be able to fill in the gaps. 0 pt
4. How would you have to extend the fragment to allow combining noun phrases with conjunctions? The goal is to support sentences like "*Ethel and the cat ran*" 0 pt

LBS WS 2024/25

Assignment7 – Modal Logics

Problem 7.1

Consider a Kripke model, where the worlds are the natural numbers (including 0) and a is accessible from b if $a \geq b$. Furthermore, let A, B, C, and D be propositional variables, where

1. A holds in a world w iff w is even,
2. B holds in w iff w is odd,
3. C only holds in world 5, and
4. D only holds in world 8.

Which of the following formulas are valid in this Kripke model? Enter `V` for valid and `I` for invalid.

1. $C \Rightarrow A$ `I`
2. $C \Rightarrow B$ `V`
3. $C \Rightarrow D$ `I`
4. $C \Rightarrow \Box A$ `I`
5. $C \Rightarrow \Box B$ `I`
6. $C \Rightarrow \Box D$ `I`
7. $C \Rightarrow \Diamond A$ `V`
8. $C \Rightarrow \Diamond B$ `V`
9. $C \Rightarrow \Diamond D$ `V`
10. $C \Rightarrow \Diamond\Diamond D$ `V`
11. $C \Rightarrow \Diamond\Box D$ `I`
12. $C \Rightarrow \Box\Diamond D$ `I`
13. $C \Rightarrow \Box\Box D$ `I`
14. $\Diamond A$ `V`
15. $\Box A$ `I`

16. $\Diamond C$ I

17. $\Box C$ I

18. $\Diamond C \Rightarrow \Diamond D$ V

19. $\Diamond D \Rightarrow \Diamond C$ I

20. $\Box C \Rightarrow \Diamond D$ V

21. $\Diamond C \Rightarrow \Box D$ I

Problem 7.2

In (propositional) modal logic, there is a correspondence between Kripke models and the axioms of the logic. For example, if the accessibility relation is reflexive and transitive, then the the following axiom is valid (among others):

$$\Box A \Rightarrow \Box\Box A$$

1. Demonstrate that the axiom is not valid if the accessibility relation is not transitive. 0 pt

---

Hint: Provide a Kripke model as a counterexample.

Solution: Consider a the Kripke model with three worlds w_1, w_2, and w_3 and the accessibility relation $\{(w_1, w_2), (w_2, w_3), (w_1, w_1), (w_2, w_2), (w_3, w_3)\}$, which is reflexive but not transitive. Let A only hold in w_1 and w_2. Then $\Box A$ holds in w_1, but $\Box\Box A$ does not.

---

2. Demonstrate that the axiom must be valid for any Kripke model with a reflexive and transitive accessibility relation. 0 pt

---

Solution: aRb denote that world b is accessible from world a. Let us assume that $\Box A$ holds in a world w. We need to show that $\Box\Box A$ also holds in w, which means that we need to show that in every world w' where wRw', $\Box A$ holds, which means that we have to show that A holds in all worlds w'' where $w'Rw''$. Now, due to transitivity, we now that wRw''. As we have assumed that $\Box A$ holds in w, A holds in all worlds w'' where $w'Rw''$.

---

3. Is reflexivity necessary for the axiom to be valid or is transitivity alone sufficient? 0 pt

---

Solution: Transitivity alone is sufficient. The solution to the previous subproblem does not depend on reflexivity.

LBS WS 2024/25

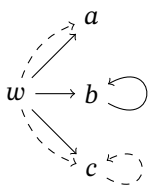
Assignment8 – Multi-Modal Logics

Problem 8.1 5 pt

Given a multimodal logic with two modalities [1] and [2]. Evaluate the following formulae

1. $[1]X$,
2. $\langle 2\rangle X$,
3. $[1]\langle 2\rangle X$,
4. $\langle 1\rangle[2]X$,
5. $\langle 1\rangle(\neg X \wedge \neg\langle 1\rangle X)$.

in world w in the following Kripke structure and briefly justify your answer. The solid arrows represent the accessibility relation for [1] and the dashed ones for [2]. Use the variable assignment φ with



$$\begin{aligned} \varphi(w,X) &= \mathsf{T} \\ \varphi(a,X) &= \mathsf{T} \\ \varphi(b,X) &= \mathsf{F} \\ \varphi(c,X) &= \mathsf{F} \end{aligned}$$

Solution: The correct answers (without justifications) are:

1. $\mathcal{I}^w_\varphi([1]X) = \mathsf{F}$,
2. $\mathcal{I}^w_\varphi(\langle 2\rangle X) = \mathsf{T}$,
3. $\mathcal{I}^w_\varphi([1]\langle 2\rangle X) = \mathsf{F}$,
4. $\mathcal{I}^w_\varphi(\langle 1\rangle[2]X) = \mathsf{T}$,
5. $\mathcal{I}^w_\varphi(\langle 1\rangle(\neg X \wedge \neg\langle 1\rangle X)) = \mathsf{T}$.

Problem 8.2 (DPL)

Which of the following expressions are valid in DL[1]?
Justify your answer – you will not get points without an explanation.

1. $[X := 3; X := 5]X = 5$ 1.5 pt

Solution:

The program $X := 3; X := 5$ (deterministically) assigns first 3 and then 5 to the variable X, so X is 5 afterwards.

---

2. $[X := 3 \cup X := 5]X = 5$ 1.5 pt

Solution: The program $X := 3 \cup X := 5$ non-deterministically assigns either 3 or 5 to the variable X, so $X = 3$ is also a possibility.

---

3. $[X := 3; X := 5; (X = 3)?]X = 5$ 1.5 pt

Solution: The program $X := 3; X := 5; (X = 3)?$ first assigns 3 to X, then 5, and then tests if X is 3. Since it is not, the program terminates. Therefore, no execution of the program terminates and it is vacuously true that $X = 5$ after every terminating execution.

---

4. $[(X := 3 \cup X := 5); (X = 3)?]X = 5$ 1.5 pt

Solution: The program $(X := 3 \cup X := 5); (X = 3)?$ non-deterministically assigns either 3 or 5 to the variable X, and then truncates all executions where X is not 3. In the $X := 3$ branch, the program terminates and $X = 5$ does not hold. Therefore, the formula is not valid.
