Free University of Bozen-Bolzano Faculty of Computer Science
R. Bernardi (teacher)

Assignment: Computational Linguistics
13/01/05

NAME:
STUDENT NUMBER:
COURSE:
YEAR:
SIGNATURE:

Assignment marking overview

Note, if you have attended the LCT colloquia you can skip the first exercise.

Marks will follow the distribution below.

[Exercise 1:	10 marks]
Exercise 2:	4 marks
Exercise 3:	13 marks
Exercise 4:	13 marks
Total:	30 marks/ [40 marks]

Note, it is required to write down the answers in a very precise way, and in all formal details. Please, attach this cover sheet to your answers.

---

1) General [10 Marks]

(At the exam, here there will be general questions on any of the topic discussed in class, for which you might be asked to give a short description of the topic itself, the main challenges and techniques, and express your opinion on what you have learned about it.)

---

2) Morphology [4 Marks]

In English, any verb ending in *-ize* can be followed by the nominalizing suffixes *-ation* or *-er*. E.g. fossilize, fossilization. Note that the root of the verb is itself a noun (fossil). Similarly, from other nouns one can derive adjectives and adverbs by adding first *-ful* and then *-ly*.

Build a Finite State Automata to model this fragment of English derivational morphology.

---

3) Syntax [13 Marks]

This exercise focus on the different ways verbs may subcategorize. (a) Give a CFG able to recognize the sentences below as grammatical and the ones marked with by * as ungrammatical.

1. I disappeared
2. I prefer a pizza
3. I gave you a pizza
4. You said I disappeared
5. He told me I disappeared
6. I want to leave
7. I left on Thursday
8. I left Boston in the morning
9. I traveled from Boston to New York
10. *You said me john left
11. *I disappear Boston
12. *I prefer
13. *I gave you
14. *I gave you on Thursday
15. *I gave from Boston to New York

(b) Build the syntactic tree for each of them following the CFG rules you have found.

4) Semantics [13 Marks]

(a) Give the lexical terms marked by their types for the words in the sentences below and (b) use the lambda-calculus to build compositionally the meaning representation of the following sentences.

1. I disappeared
2. I prefer a pizza
3. I gave you a pizza
4. You said I disappeared
5. Every student disappeared
6. Every student passed an exam

1 Solutions

1.1 Exercise 1

1.2 Exercise 2

```
s --> np vp
np --> det n
np --> pn
vp --> iv
vp --> tv np
vp --> dtv np np
vp --> vs s
vp --> vns np s
vp --> vi inf
inf --> pt i
vp --> iv np pp
vp --> iv pp
ppt --> pto inf
vp --> iv' pp' pp'
pp --> p np

iv --> disappeared
iv --> left
tv --> prefer
dtv --> gave
vs --> said
vns --> told
vi --> want
iv' --> traveled
i --> leave
pt --> to
pn --> I
```

(b)

1. $[[[I]_{pn}]_{np}[[disappeared]_{iv}]_{vp}]_s$
2. $[[[I]_{pn}]_{np}[[prefer]_{tv}[[a]_{det}[pizza]_n]_{np}]_{vp}]_s$
3. $[[[I]_{pn}]_{np}[[gave]_{dtv}[you]_{np}[[a]_{det}[pizza]_n]_{np}]_{vp}]_s$
4. $[[[You]_{pn}]_{np}[[said]_{vs}[[[I]_{pn}]_{np}[disappeared]_{iv}]_{vp}]_s]_s$
5. $[[[I]_{pn}]_{np}[[told]_{vns}[[me]_{pn}]_{np}[[I]_{np}[disappeared]_{iv}]_{vp}]_s]_s$
6. $[[[I]_{pn}]_{np}[[want]_{vi}[[to]_{pt}[leave]_i]_{inf}]_{vp}]_s$
7. $[[[I]_{pn}]_{np}[[left]_{iv}[[on]_p[Thursday]_{np}]_{pp}]_{vp}]_s$
8. $[[[I]_{pn}]_{np}[[left]_{iv}[Boston]_{np}[[in]_p[[the]_{det}[morning]_n]_{np}]_{pp}]_{vp}]_s$
9. $[[[I]_{pn}]_{np}[[traveled]_{iv'}[[from]_p[Boston]_{np}]_{pp}[[to]_p[NewYork]_{np}]_{pp'}]_{vp}]_s$
10. *You said me john left
11. *I disappear Boston
12. *I prefer
13. *I gave you
14. *I gave you on Thursday
15. *I gave from Boston to New York

Note, this grammar will generate trees which are no-binary (e.g. 3.)

1.3 Exercise 4

(a) Lexical terms:

I	:=	$\mathtt{i}$
you	:=	$\mathtt{y}$
student	:=	$\lambda x.\mathtt{Student}(x)$
pizza	:=	$\lambda x.\mathtt{Pizza}(x)$
exam	:=	$\lambda x.\mathtt{Exam}(x)$
disappear	:=	$\lambda x.\mathtt{Disappear}(x)$
prefer	:=	$\lambda x.\lambda y.\mathtt{Prefer}(y, x)$
prefer	:=	$\lambda x.\lambda y.\mathtt{Passed}(y, x)$
gave	:=	$\lambda z.\lambda x.\lambda y.\mathtt{Student}(y, x, z)$
an/a	:=	$\lambda X.\lambda Y.\exists x.X(x) \wedge Y(x)$
every	:=	$\lambda X.\lambda Y.\forall x.X(x) \rightarrow Y(x)$

(b)

1. $\mathtt{Disappear}(\mathtt{i})$
2. $\exists x.\mathtt{Pizza}(x) \wedge \mathtt{Prefer}(\mathtt{i}, x)$
3. $\exists x.\mathtt{Pizza}(x) \wedge \mathtt{Gave}(\mathtt{i}, x, \mathtt{y})$
4. $\mathtt{Said}(\mathtt{y}, \mathtt{Disappear}(\mathtt{i}))$
5. $\forall x.\mathtt{Student}(x) \rightarrow \mathtt{Disappear}(x)$
6. $\forall x.\mathtt{Student}(x) \rightarrow \exists z.\mathtt{Exam}(z) \wedge \mathtt{Passed}(x, z)$

7. $\exists z.\texttt{Exam}(z) \rightarrow \forall x.\texttt{Student}(x) \wedge \texttt{Passed}(x, z)$

I give the solution of the last sentence by means of example (the others are easier). (Note, I skip some steps, you have to write down all of them)

(6a)

- an exam: $\lambda Y.\exists x.\texttt{Exam}(x) \wedge Y(x)$
- Every student: $\lambda Y.\forall x.\texttt{Student}(x) \rightarrow Y(x)$
- u passed y': $\texttt{Passed}(u, y')$
- u passed: $\lambda y'.\texttt{Passed}(u, y')$
- u passed an exam: $\exists x.\texttt{Exam}(x) \wedge \texttt{Passed}(u, x)$
- passed an exam: $\lambda u.\exists x.\texttt{Exam}(x) \wedge \texttt{Passed}(u, x)$
- Every student passed an exam: $\forall x.\texttt{Student}(x) \rightarrow \exists z.\texttt{Passed}(x, z)$

(6b)

- an exam: $\lambda Y.\exists x.\texttt{Exam}(x) \wedge Y(x)$
- every student: $\lambda Y.\forall x.\texttt{Student}(x) \rightarrow Y(x)$
- passed u: $\lambda y.\texttt{Passed}(y, u)$
- every student passed u: $\forall x.\texttt{Student}(x) \wedge \texttt{Passed}(x, u)$
- every student passed: $\lambda u.\forall x.\texttt{Student}(x) \wedge \texttt{Passed}(x, u)$
- Every student passed an exam: $\exists z.\texttt{Exam}(z) \wedge \forall x.\texttt{Student}(x) \rightarrow \texttt{Passed}(x, z)$