Free University of Bozen-Bolzano Faculty of Computer Science

R. Bernardi

Sample Exam: Computational Linguistics

NAME:
STUDENT NUMBER:
COURSE:
YEAR:
SIGNATURE:

Assignment marking overview

This assignment will constitute the 50 % of the whole exam.

Marks will follow the distribution below.

Exercise 1:	10 marks
Exercise 2:	15 marks
Exercise 3:	25 marks
Total:	50 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) Syntax (CFG) [10 Marks]

This exercise focuses on the different ways verbs may subcategorize. Given the CFG below build parse trees for the following grammatical sentences (you could use brackets notation if you like) and check that the ungrammatical ones are not recognized, if they are change the grammar (make sure that the changes still generate the grammatical ones, though). [Note, at the exam, I might ask you to provide the rules your self]

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

CFG rules:

```
s --> np vp
np --> det n
np --> pn
vp --> iv
vp --> tv np
vp --> dtv np np
vp --> vns 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
vns --> said
vns --> told
vi --> want
iv --> traveled
i --> leave
pto --> to
pn --> I
pn -->  you
det --> a
n --> pizza
p --> on
p --> from
p --> to
np --> Thursday
```

---

2) Semantics (lambda terms) [15 Marks]

(a) Give the lexical lambda terms marked by their types for the words in the sentences below and (b) use the lambda-calculus to build compositionaly 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

---

3) Syntax-Semantics interface (CG and lambda terms) [25 Marks]

(a) Find CG categories to be assigned to the lexical items in the sentence "Every student who met Mary left" and build the CG tree for it. Do you need to use both elimination and introduction rules? Comment on their use with respect of the sentence we are considering. [10 marks]

(b) Give typed lambda terms to the lexicon entries and verify that their types correspond to the CG categories (accordingly to the standard mapping). Check that the lambda terms are properly representing the set-theoretical meaning of the corresponding words by giving the latter too. [15 marks]

1 Solutions

1.1 Exercise 1

Modified grammar:

```
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
pp' --> p' np

iv --> disappeared
iv --> left
tv --> prefer
dtv --> gave
```

```
vs --> said
vns --> told
vi --> want
iv' --> traveled
i --> leave
pto --> to
pn --> I
pn -->  you
det --> a
n --> pizza
p --> on
p' --> from
p' --> to
np --> Thursday
```

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. $[[[I]_{pn}]_{np}[[traveled]_{iv'}[[from]_{p'}[Boston]_{np}]_{pp'}[[to]_{p'}[NewYork]_{np}]_{pp'}]_{vp}]_s$

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

1.2 Exercise 2

(a) Lexical terms:

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

(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. $\texttt{Said}(\texttt{y}, \texttt{Disappear}(\texttt{i}))$
5. $\forall x.\texttt{Student}(x) \rightarrow \texttt{Disappear}(x)$

6a. $\forall x.\texttt{Student}(x) \rightarrow \exists z.\texttt{Exam}(z) \wedge \texttt{Passed}(x, z)$

6b. $\exists z.\texttt{Exam}(z) \rightarrow \forall x.\texttt{Student}(x) \wedge \texttt{Passed}(x, z)$

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

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

1.3 Exercise 3

Every	$s/(np\backslash s)$	$\lambda X.\lambda Y.\forall x.X(x) \rightarrow Y(x)$	$(e \rightarrow t) \rightarrow ((e \rightarrow t) \rightarrow t)$
student	n	$\lambda y.\texttt{Student}(y)$	$e \rightarrow t$
who	$(n\backslash n)/(s/np)$	$\lambda X.\lambda Y.\lambda y.Y(y) \wedge X(y)$	$(e \rightarrow t) \rightarrow ((e \rightarrow t) \rightarrow (e \rightarrow t))$
mary	np	$\texttt{m}$	e
met	$(np\backslash s)/np$	$\lambda x.\lambda y.\texttt{met}(y, x)$	$(e \rightarrow (e \rightarrow t$
left	$np\backslash s$	$\lambda y.\texttt{left}(x)$	$(e \rightarrow t)$

- met u: $\lambda y.\texttt{met}(y, u) : np\backslash s$
- mary met u: $\texttt{met}(\texttt{m}, u) : s$
- mary met: $\lambda u.\texttt{met}(\texttt{m}, u) : s/np$
- who mary met: $\lambda Y.\lambda y.Y(y) \wedge \texttt{met}(\texttt{m}, y) : n\backslash n$
- student who mary met: $\lambda y.\texttt{Student}(y) \wedge \texttt{met}(\texttt{m}, y) : n$
- Every student who mary met: $\lambda Y.\forall x.(\texttt{Student}(x) \wedge \texttt{met}(\texttt{m}, x)) \rightarrow Y(x) : s/(np\backslash s)$
- Every student who mary met left: $\forall x.(\texttt{Student}(x) \wedge \texttt{met}(\texttt{m}, x)) \rightarrow \texttt{left}(x) : s$

You should make some comment here regarding the rules of CG you need to apply. Observe steps 2 and 3.

Furthermore, you should comment on why the lambda terms given represent the meaning of the corresponding words correctly (ie. by illustrating their set theoretical interpretation and the link from sets to functions, i.e. types. Plus, you should comment on the connection between types and syntactic categories.