LING 388: Language and Computers

Sandiway Fong

Lecture 3
Administrivia

• Today’s Topics
  1. Homework 1 Review
  2. More on the Stanford Parser
  3. Introduction to Prolog

  – Homework 2: Install SWI-Prolog on your laptop
Homework 1 Review

Examine the Stanford Parser output on these two sentences:

1. Which car did Mary like?
2. *Which car did Mary like the convertible?

Questions:
1. (4pts) Does the typed dependencies output offer any advantage over the parse output for the 1st sentence?
Homework 1 Review

Examine the Stanford Parser output on these two sentences:

1. Which car did Mary like?
2. *Which car did Mary like the convertible?

Questions:
2. (4pts) What is wrong with the 2\textsuperscript{nd} sentence?

Diagram:

- **like**: agent, theme
- Mary
- the convertible
- (which) car

?
Homework 1 Review

Examine the Stanford Parser output on these two sentences:

1. Which car did Mary like?
2. *Which car did Mary like the convertible?

Questions:

3. (4pts) Is it possible to deduce from the Stanford Parser output that the 2\textsuperscript{nd} sentence is ungrammatical?
Homework 1 Review

• Recall the ambiguous example:
  – *Where can I see the bus stop?*
  
the Stanford parser analyses “*bus stop*” preferentially as a
noun-noun compound:

(NP (DT the) (NN bus) (NN stop))

4. (4pts) Give a (question) sentence ending in “… *the bus stop*?” where the Stanford parser analyses “*stop*” as a verb.

5. (4pts) What syntactic situations would force a parser to
decide to analyze “*stop*” in “… *the bus stop*?” as a noun
(vs. a verb)?
Homework 1 Review

• Recall the ambiguous example:
  – Where can I see the bus stop?
    the Stanford parser analyses “bus stop” preferentially as a noun-noun compound:
      (NP (DT the) (NN bus) (NN stop))

4. (4pts) Give a (question) sentence ending in “… the bus stop?” where the Stanford parser analyses “stop” as a verb.

Parse

(ROOT
  (SQ (VBD Did)
    (NP (DT the) (NN bus))
    (VP (VB stop)))
  (. ?)))
Homework 1 Review

• Recall the ambiguous example:
  – *Where can I see the bus stop?*
  
  the Stanford parser analyses "bus stop" preferentially as a noun-noun compound:
  
  \[(NP (DT the) (NN bus) (NN stop))\]

5. (4pts) What syntactic situations would force a parser to decide to analyze "stop" in "... the bus stop?" as a noun (vs. a verb)?

In English, verbs don’t normally appear at the end of the sentence. To end in an uninflected verb (*stop*), we can form a question...

1. yes/no question
2. object *wh*-question
Your query

*which protesters did the police stop?*

Tagging

*which/WDT protesters/NN did/VBD the/DT police/NN stop/VB*

Parse

(ROOT
  (SBARQ
    (WHNP (WDT which) (NN protesters))
    (SQ (VBD did)
      (NP (DT the) (NN police))
      (VP (VB stop))
    )
  )
)

Typed dependencies

det(protesters-2, which-1)
dobj(stop-6, protesters-2)
aux(stop-6, did-3)
det(policing-5, the-4)
nsubj(stop-6, police-5)
root(ROOT-0, stop-6)
Homework 1 Review

Can you let me know where will the bus stop?

Tagging
Can/MD you/PRP let/VB me/PRP know/VBP where/WRB will/MD the/DT bus/NN stop/VB ?/.

Parse
(ROOT
  (SQ (MD Can))
  (NP (PRP you))
  (VP (VB let))
  (SBAR
    (S
      (NP (PRP me))
      (VP (VBP know))
      (SBARQ
        (WHADVP (WRB where))
        (SQ (MD will))
        (NP (DT the) (NN bus))
        (NP (DT the) (NN bus))
        (VP (VB stop)))))
  (. ?)))

maybe a bit awkward ...
Cf. Can you let me know where the bus will stop?
Stanford Parser

Your query

John was arrested by the police.

Tagging

John/NNP was/VBD arrested/VBN by/IN the/DT police/NN

Parse

(ROOT
  (S
    (NP (NNP John))
    (VP (VBD was)
      (VP (VBN arrested)
        (PP (IN by)
          (NP (DT the) (NN police))))))
  (. .)))
Typed dependencies

nsbjpass(arrested-3, John-1)
auxpass(arrested-3, was-2)
root(ROOT-0, arrested-3)
prep(arrested-3, by-4)
det(policeman-6, the-5)
pobj(by-4, police-6)

Typed dependencies, collapsed

nsbjpass(arrested-3, John-1)
auxpass(arrested-3, was-2)
root(ROOT-0, arrested-3)
det(policeman-6, the-5)
agent(arrested-3, police-6)
Typed dependencies

nsubjpass(arrested-3, John-1)
auxpass(arrested-3, was-2)
root(ROOT-0, arrested-3)
prep(arrested-3, by-4)
det(police-6, the-5)
pobj(by-4, police-6)

Typed dependencies, collapsed

nsubjpass(arrested-3, John-1)
auxpass(arrested-3, was-2)
root(ROOT-0, arrested-3)
det(police-6, the-5)
agent(arrested-3, police-6)
SWI-Prolog

• AME S314
  – I’ve requested that SWI Prolog be installed on the iMac workstations

• Computation based on logic and proceeding via inference

• Computer language we’ll be using:
  – Name: PROLOG (PROgramming in LOGic)
  – Variant: SWI-PROLOG (free software)
  – Free download: http://www.swi-prolog.org/
  – Designed to express logic statements and phrase structure grammar rules
SWI-Prolog

Your homework:
• Install SWI-Prolog on your laptop
• Read about Prolog online
Prolog online resources

- Some background in logic or programming?
- Useful Online Tutorials
  - Learn Prolog Now!
    - Patrick Blackburn, Johan Bos & Kristina Striegnitz
    - http://www.learnprolognow.org
  - An introduction to Prolog
    - Michel Loiseleur & Nicolas Vigier
    - http://boklm.eu/prolog/page_0.html
SWI-Prolog

- Next time, we’ll look at grammars and how to represent rules in Prolog. We already saw a preview of a Prolog grammar in lecture 1:

```
s -> np, vp.
v -> v, np.
np -> d, n.
d -> [the].
n -> [man].
n -> [cheese].
v -> [ate].
```

- Today:
  - introduction to more basic Prolog concepts
SWI-Prolog

• Prolog is different from most programming languages
  • most languages:
    – think sequentially: first do A then B, loop around ...
    – call the program
  • Prolog:
    – there’s a database
    – state facts
    – state rules
    – run: ask a question (query)
SWI-Prolog

- Starting Prolog:
  - swipl (on Linux)
  - /opt/local/bin/swipl (on Macs)
SWI-Prolog

- Starting Prolog:
  - swipl (on Linux)
  - /opt/local/bin/swipl (on Macs)

```
load file test.pl from current directory
```

```
For help, use ?- help(Topic). or ?- apropos(Word).

?- listing.

:- thread_local thread_message_hook/3.
:- dynamic thread_message_hook/3.
:- volatile thread_message_hook/3.

true.

?- [test].
ERROR: source_sink 'test' does not exist
?- 
```

show what’s currently in the database
load file test.pl from current directory
SWI-Prolog

• Adding facts/rules:
  – directly from the Prolog command line: assert(rule).
  – from a file: consult(file). or [file].
  – default extension: .pl (file.pl)
SWI-Prolog

Let’s look at this fact/rule/query programming paradigm

• Chapter 1 of Learn Prolog Now

**Chapter 1**
**Facts, Rules, and Queries**

This chapter has two main goals:

1. To give some simple examples of Prolog programs. This will introduce us to the three basic constructs in Prolog: facts, rules, and queries. It will also introduce us to a number of other themes, like the role of logic in Prolog, and the idea of performing unification with the aid of variables.
2. To begin the systematic study of Prolog by defining terms, atoms, variables and other syntactic concepts.

1.1 Some Simple Examples
   Knowledge Base 1
   Knowledge Base 2
   Knowledge Base 3
   Knowledge Base 4
   Knowledge Base 5

1.2 Prolog Syntax
   Atoms
   Numbers
   Variables
   Complex terms

1.3 Exercises

1.4 Practical Session
?- working_directory(X,X).
X = '/Users/sandiway/Desktop/'.

?- working_directory(X, '/Users/sandiway').
X = '/Users/sandiway/Desktop/'.

?- working_directory(X,X).
X = '/Users/sandiway/'.

?- |