Simple parser – in-class mini-project

Semantics Seminar: Computing Dynamic Meanings, Spring 2015, UCSC

Nathan Arnett, Karl DeVries, Karen Duek, Kelsey Kraus,
Veronika Richtarcikova, Deniz Rudin, Adrian Brasoveanu*

April 28, 2015

Contents

This pdf file contains the code and simulations for our ongoing in-class mini-project, which is putting together a simple parser (at least for now).

1 The ACT-R model: A mostly top-down parser

We see that when we reach the `expand_VP` and `expand_VP_Vt` rules (the first one expands the VP hypothesizing an intransitive verb, the second one expands a VP hypothesizing a transitive verb), we might end up with a declarative memory (DM) error depending on whether the scanned verb is transitive or not. In that case, the `dm_error` rule is triggered and the entire parse is restarted. We try again and again until we end up choosing the correct rule for VP expansion.

This could be enhanced by keeping track of choice points and restarting only part of the parse (i.e., we backtrack to the closest choice point).

(1) File **simple_parser.py**:

```
import ccm
from ccm.lib.actr import *
from collections import deque


class MotorModule(ccm.Model):
    def read_next_word(self):
        self.parent.parent.current_wd_pos += 1
        return self.parent.parent.sent[self.parent.parent.current_wd_pos]
```

*We want to thank Jakub Dotlačil for various general ideas and code related to the implementation of parsers in Python ACT-R. The authors are listed in alphabetical order, with the instructor listed last.

```
class MyAgent(ACTR):
    goal = Buffer()
    goalstack = deque()

    DMBuffer = Buffer()
    DM = Memory(DMBuffer)

    motor = MotorModule()
    motorBuffer = Buffer()

    def init():
        DM.add('string:Bob cat:NP')
        DM.add('string:Mary cat:NP')
        DM.add('string:sleeps cat:V')
        DM.add('string:likes cat:Vt')
        DM.add('string:the cat:D')
        DM.add('string:bank cat:N')
        DM.add('string:was cat:Copula')
        DM.add('string:closed cat:adj')
        DM.add('string:mushy cat:adj')
        goal.set("parse-ongoing")

    def root_S(goal="parse-ongoing"):
        DM.add('string:None cat:S pos:0')
        goalstack.append('string:None cat:S pos:0')
        DM.add('string:None cat:VP pos:01')
        goalstack.append('string:None cat:VP pos:01')
        DM.add('string:None cat:NP pos:00')
        goalstack.append('string:None cat:NP pos:00')
        goal.set("scan-word")

    def scan_word(goal="scan-word"):
        word = motor.read_next_word()
        if word != "END":
            motorBuffer.set('string:?word')
            goal.set(goalstack.pop())
        else:
            goal.set("parse-END")

    def request_NP(goal='string:None cat:NP pos:?pos', \
                   motorBuffer='string:?word', \
                   DMBuffer=None):
        DM.request('string:?word cat:NP')

    def parse_NP(goal='string:None cat:NP pos:?pos', \
                 motorBuffer='string:?word', \
                 DMBuffer="string:?word cat:NP"):
        pos_word = pos + '0'
```

```
    DM.add('string:?word cat:?word pos:?pos_word')
    motorBuffer.clear()
    DMBuffer.clear()
    goal.set("scan-word")
    self.draw_tree()

#def request_VP(goal='string:None cat:VP pos:?pos', \
                #motorBuffer='string:?word', \
                #DMBuffer=None):
    #DM.request('string:?word cat:?cat')

#def parse_VP(goal='string:None cat:VP pos:?pos', \
              #motorBuffer='string:?word', \
              #DMBuffer="string:?word cat:V"):
    #pos_V = pos + "0"
    #pos_word = pos_V + '0'
    #DM.add('string:None cat:V pos:?pos_V')
    #DM.add('string:?word cat:?word pos:?pos_word')
    #motorBuffer.clear()
    #DMBuffer.clear()
    #goal.set("scan-word")
    #self.draw_tree()

def expand_VP(goal='string:None cat:VP pos:?pos', \
              DMBuffer=None):
    goalstack.append(goal.chunk)
    pos_V = pos + "0"
    DM.add('string:None cat:V pos:?pos_V')
    goal.set('string:None cat:V pos:?pos_V')

def expand_VP_Vt(goal='string:None cat:VP pos:?pos', \
                 DMBuffer=None):
    goalstack.append(goal.chunk)
    pos_NP = pos + "1"
    DM.add('string:None cat:NP pos:?pos_NP')
    goal.set('string:None cat:NP pos:?pos_NP')
    goalstack.append(goal.chunk)
    pos_Vt = pos + "0"
    DM.add('string:None cat:Vt pos:?pos_Vt')
    goal.set('string:None cat:Vt pos:?pos_Vt')

def request_V(goal='string:None cat:V pos:?pos', \
              motorBuffer='string:?word', \
              DM='error:False', \
              DMBuffer=None):
    DM.request('string:?word cat:V')

def parse_V(goal='string:None cat:V pos:?pos', \
            motorBuffer='string:?word', \
            DMBuffer="string:?word cat:V"):
```

```
    pos_word = pos + '0'
    DM.add('string:?word cat:?word pos:?pos_word')
    motorBuffer.clear()
    DMBuffer.clear()
    goal.set("scan-word")
    self.draw_tree()

def request_Vt(goal='string:None cat:Vt pos:?pos', \
               motorBuffer='string:?word', \
               DM='error:False', \
               DMBuffer=None):
    DM.request('string:?word cat:Vt')

def parse_Vt(goal='string:None cat:Vt pos:?pos', \
             motorBuffer='string:?word', \
             DMBuffer="string:?word cat:Vt"):
    pos_word = pos + '0'
    DM.add('string:?word cat:?word pos:?pos_word')
    motorBuffer.clear()
    DMBuffer.clear()
    goal.set("scan-word")
    self.draw_tree()

def dm_error(DM='error:True'):
    goal.clear()
    goalstack.clear()
    DMBuffer.clear()
    self.parent.current_wd_pos = -1
    goal.set("parse-ongoing")

def stop_production(goal="parse-END"):
    goal.clear()
    goalstack.clear()
    DMBuffer.clear()
    print "FINAL TREE:"
    self.draw_tree()

def draw_tree(self):
    list_of_XPs = self.DM.find_matching_chunks('')
    list_of_XPs = [XP for XP in list_of_XPs if XP.has_key("cat") and XP.has_key("pos")]
    #print list_of_XPs
    tree_dict = {}
    for idx in range(len(list_of_XPs)):
        tree_dict[list_of_XPs[idx]["pos"]] = list_of_XPs[idx]["cat"]
    # we now generate the ASCII tree (based on code here:
    # http://stackoverflow.com/questions/10879063/parsing-a-list-of-words-into-a-tree/11
    from ete2 import Tree
    paths = tree_dict.keys()
    # Create empty tree
    tree = Tree()
```

```
        tree.name = ""
        # Keep tree structure indexed
        name2node = {}
        # No duplicates
        paths = set(paths)
        # Populate tree
        for wd in paths:
            # If no similar paths exist, add it to the base of tree
            target = tree
            # Find relatives in the tree
            for pos in xrange(len(wd), -1, -1):
                root = wd[:pos]
                if root in name2node:
                    target = name2node[root]
                    break
            # Add new nodes as necessary
            fullname = root
            for letter in wd[pos:]:
                fullname += letter
                new_node = target.add_child(name=tree_dict[fullname], dist=1.0)
                name2node[fullname] = new_node
                target = new_node
        # Print tree
        print tree.get_ascii()
        print ""


# previously known as environment
class Sentence(ccm.Model):
    def __init__(self, input_sent=None):
        ccm.Model.__init__(self)
        self.sent = input_sent.split() + ["END"]
        self.current_wd_pos = -1


if __name__ == "__main__":
    #example = "Bob sleeps"
    example = "Bob likes Mary"
    #example = "the bank was closed"
    #example = "the bank was mushy"
    simulation = Sentence(example)
    simulation.agent = MyAgent()
    ccm.log_everything(simulation)
    #log=ccm.log(html=True)
    simulation.run(5)
    ccm.finished()
```

2 Parsing *Bob sleeps*

[py1]
```
>>> from simple_parser import *
>>> example = "Bob sleeps"
>>> simulation = Sentence(example)
>>> simulation.agent = MyAgent()
>>> ccm.log_everything(simulation)
   0.000 current_wd_pos -1
   0.000 agent.production_threshold None
   0.000 agent.production_time_sd None
   0.000 agent.production_match_delay 0
   0.000 agent.production_time 0.05
   0.000 agent.DM.record_all_chunks False
   0.000 agent.DM.threshold 0
   0.000 agent.DM.latency 0.05
   0.000 agent.DM.busy False
   0.000 agent.DM.maximum_time 10.0
   0.000 agent.DM.error False
   0.000 agent.goal.chunk None
   0.000 agent.DMBuffer.chunk None
   0.000 agent.motorBuffer.chunk None
>>> simulation.run()
   0.000 agent.goal.chunk parse-ongoing
   0.000 agent.production root_S
   0.050 agent.production None
   0.050 agent.goal.chunk scan-word
   0.050 agent.production scan_word
   0.100 agent.production None
   0.100 current_wd_pos 0
   0.100 agent.motorBuffer.chunk string:Bob
   0.100 agent.goal.chunk cat:NP pos:00 string:None
   0.100 agent.production request_NP
   0.150 agent.production None
   0.150 agent.DM.busy True
   0.150 agent.production request_NP
   0.200 agent.DMBuffer.chunk cat:NP string:Bob
   0.200 agent.DM.busy False
   0.200 agent.production (changed before firing)
   0.200 agent.production parse_NP
   0.250 agent.production None
   0.250 agent.motorBuffer.chunk None
   0.250 agent.DMBuffer.chunk None
   0.250 agent.goal.chunk scan-word

         /NP/-Bob
-- /S|
         \-VP

   0.250 agent.production scan_word
   0.300 agent.production None
```

```
   0.300 current_wd_pos 1
   0.300 agent.motorBuffer.chunk string:sleeps
   0.300 agent.goal.chunk cat:VP pos:01 string:None
   0.300 agent.production expand_VP
   0.350 agent.production None
   0.350 agent.goal.chunk cat:V pos:010 string:None
   0.350 agent.production request_V
   0.400 agent.production None
   0.400 agent.DM.busy True
   0.400 agent.production request_V
   0.450 agent.DMBuffer.chunk cat:V string:sleeps
   0.450 agent.DM.busy False
   0.450 agent.production (changed before firing)
   0.450 agent.production parse_V
   0.500 agent.production None
   0.500 agent.motorBuffer.chunk None
   0.500 agent.DMBuffer.chunk None
   0.500 agent.goal.chunk scan-word

      /NP/-Bob
-- /S|
      \VP/V /-sleeps

   0.500 agent.production scan_word
   0.550 agent.production None
   0.550 current_wd_pos 2
   0.550 agent.goal.chunk parse-END
   0.550 agent.production stop_production
   0.600 agent.production None
   0.600 agent.goal.chunk None
   0.600 agent.DMBuffer.chunk None
FINAL TREE:

      /NP/-Bob
-- /S|
      \VP/V /-sleeps

>>> ccm.finished()
```

3 Parsing *Bob likes Mary*

[py2]
```
>>> from simple_parser import *
>>> example = "Bob likes Mary"
>>> simulation = Sentence(example)
>>> simulation.agent = MyAgent()
>>> ccm.log_everything(simulation)
   0.000 current_wd_pos -1
   0.000 agent.production_threshold None
   0.000 agent.production_time_sd None
```

```
   0.000 agent.production_match_delay 0
   0.000 agent.production_time 0.05
   0.000 agent.DM.record_all_chunks False
   0.000 agent.DM.threshold 0
   0.000 agent.DM.latency 0.05
   0.000 agent.DM.busy False
   0.000 agent.DM.maximum_time 10.0
   0.000 agent.DM.error False
   0.000 agent.goal.chunk None
   0.000 agent.DMBuffer.chunk None
   0.000 agent.motorBuffer.chunk None
>>> simulation.run()
   0.000 agent.goal.chunk parse-ongoing
   0.000 agent.production root_S
   0.050 agent.production None
   0.050 agent.goal.chunk scan-word
   0.050 agent.production scan_word
   0.100 agent.production None
   0.100 current_wd_pos 0
   0.100 agent.motorBuffer.chunk string:Bob
   0.100 agent.goal.chunk cat:NP pos:00 string:None
   0.100 agent.production request_NP
   0.150 agent.production None
   0.150 agent.DM.busy True
   0.150 agent.production request_NP
   0.200 agent.DMBuffer.chunk cat:NP string:Bob
   0.200 agent.DM.busy False
   0.200 agent.production (changed before firing)
   0.200 agent.production parse_NP
   0.250 agent.production None
   0.250 agent.motorBuffer.chunk None
   0.250 agent.DMBuffer.chunk None
   0.250 agent.goal.chunk scan-word

      /NP/-Bob
-- /S|
      \-VP

   0.250 agent.production scan_word
   0.300 agent.production None
   0.300 current_wd_pos 1
   0.300 agent.motorBuffer.chunk string:likes
   0.300 agent.goal.chunk cat:VP pos:01 string:None
   0.300 agent.production expand_VP
   0.350 agent.production None
   0.350 agent.goal.chunk cat:V pos:010 string:None
   0.350 agent.production request_V
   0.400 agent.production None
   0.400 agent.DM.busy True
   0.400 agent.production request_V
```

```
   0.450 agent.DM.error True
   0.450 agent.DMBuffer.chunk None
   0.450 agent.DM.busy False
   0.450 agent.production (changed before firing)
   0.450 agent.production dm_error
   0.500 agent.production None
   0.500 agent.goal.chunk None
   0.500 agent.DMBuffer.chunk None
   0.500 current_wd_pos -1
   0.500 agent.goal.chunk parse-ongoing
   0.500 agent.production dm_error
   0.550 agent.production None
   0.550 agent.goal.chunk None
   0.550 agent.DMBuffer.chunk None
   0.550 current_wd_pos -1
   0.550 agent.goal.chunk parse-ongoing
   0.550 agent.production dm_error
   0.600 agent.production None
   0.600 agent.goal.chunk None
   0.600 agent.DMBuffer.chunk None
   0.600 current_wd_pos -1
   0.600 agent.goal.chunk parse-ongoing
   0.600 agent.production root_S
   0.650 agent.production None
   0.650 agent.DM.error False
   0.650 agent.goal.chunk scan-word
   0.650 agent.production scan_word
   0.700 agent.production None
   0.700 current_wd_pos 0
   0.700 agent.motorBuffer.chunk string:Bob
   0.700 agent.goal.chunk cat:NP pos:00 string:None
   0.700 agent.production request_NP
   0.750 agent.production None
   0.750 agent.DM.busy True
   0.750 agent.production request_NP
   0.800 agent.DMBuffer.chunk cat:NP string:Bob
   0.800 agent.DM.busy False
   0.800 agent.production (changed before firing)
   0.800 agent.production parse_NP
   0.850 agent.production None
   0.850 agent.motorBuffer.chunk None
   0.850 agent.DMBuffer.chunk None
   0.850 agent.goal.chunk scan-word

      /VP/-V
-- /S|
      \NP/-Bob

   0.850 agent.production scan_word
   0.900 agent.production None
```

```
   0.900 current_wd_pos 1
   0.900 agent.motorBuffer.chunk string:likes
   0.900 agent.goal.chunk cat:VP pos:01 string:None
   0.900 agent.production expand_VP
   0.950 agent.production None
   0.950 agent.goal.chunk cat:V pos:010 string:None
   0.950 agent.production request_V
   1.000 agent.production None
   1.000 agent.DM.busy True
   1.000 agent.production request_V
   1.050 agent.DM.error True
   1.050 agent.DMBuffer.chunk None
   1.050 agent.DM.busy False
   1.050 agent.production (changed before firing)
   1.050 agent.production dm_error
   1.100 agent.production None
   1.100 agent.goal.chunk None
   1.100 agent.DMBuffer.chunk None
   1.100 current_wd_pos -1
   1.100 agent.goal.chunk parse-ongoing
   1.100 agent.production dm_error
   1.150 agent.production None
   1.150 agent.goal.chunk None
   1.150 agent.DMBuffer.chunk None
   1.150 current_wd_pos -1
   1.150 agent.goal.chunk parse-ongoing
   1.150 agent.production root_S
   1.200 agent.production None
   1.200 agent.DM.error False
   1.200 agent.goal.chunk scan-word
   1.200 agent.production scan_word
   1.250 agent.production None
   1.250 current_wd_pos 0
   1.250 agent.motorBuffer.chunk string:Bob
   1.250 agent.goal.chunk cat:NP pos:00 string:None
   1.250 agent.production request_NP
   1.300 agent.production None
   1.300 agent.DM.busy True
   1.300 agent.production request_NP
   1.350 agent.DMBuffer.chunk cat:NP string:Bob
   1.350 agent.DM.busy False
   1.350 agent.production (changed before firing)
   1.350 agent.production parse_NP
   1.400 agent.production None
   1.400 agent.motorBuffer.chunk None
   1.400 agent.DMBuffer.chunk None
   1.400 agent.goal.chunk scan-word

      /VP/-V
-- /S|
```

```
      \NP/-Bob

   1.400 agent.production scan_word
   1.450 agent.production None
   1.450 current_wd_pos 1
   1.450 agent.motorBuffer.chunk string:likes
   1.450 agent.goal.chunk cat:VP pos:01 string:None
   1.450 agent.production expand_VP_Vt
   1.500 agent.production None
   1.500 agent.goal.chunk cat:NP pos:011 string:None
   1.500 agent.goal.chunk cat:Vt pos:010 string:None
   1.500 agent.production request_Vt
   1.550 agent.production None
   1.550 agent.DM.busy True
   1.550 agent.production request_Vt
   1.600 agent.DMBuffer.chunk cat:Vt string:likes
   1.600 agent.DM.busy False
   1.600 agent.production (changed before firing)
   1.600 agent.production parse_Vt
   1.650 agent.production None
   1.650 agent.motorBuffer.chunk None
   1.650 agent.DMBuffer.chunk None
   1.650 agent.goal.chunk scan-word

      /NP/-Bob
-- /S|
     |   /Vt/-likes
      \VP
         \-NP

   1.650 agent.production scan_word
   1.700 agent.production None
   1.700 current_wd_pos 2
   1.700 agent.motorBuffer.chunk string:Mary
   1.700 agent.goal.chunk cat:NP pos:011 string:None
   1.700 agent.production request_NP
   1.750 agent.production None
   1.750 agent.DM.busy True
   1.750 agent.production request_NP
   1.800 agent.DMBuffer.chunk cat:NP string:Mary
   1.800 agent.DM.busy False
   1.800 agent.production (changed before firing)
   1.800 agent.production parse_NP
   1.850 agent.production None
   1.850 agent.motorBuffer.chunk None
   1.850 agent.DMBuffer.chunk None
   1.850 agent.goal.chunk scan-word

      /NP/-Bob
-- /S|
```

```
      |   /NP/-Mary
       \VP
          \Vt/-likes

   1.850 agent.production scan_word
   1.900 agent.production None
   1.900 current_wd_pos 3
   1.900 agent.goal.chunk parse-END
   1.900 agent.production stop_production
   1.950 agent.production None
   1.950 agent.goal.chunk None
   1.950 agent.DMBuffer.chunk None
FINAL TREE:

      /NP/-Bob
-- /S|
      |   /NP/-Mary
       \VP
          \Vt/-likes

>>> ccm.finished()
```