Patrick Herron
Tool I
Due 05 October 2004

Summary

The author of the present study created and used ten sentences for the purpose of testing the performance of three different language parsers. Sentences were chosen on the basis of a set of features typically troublesome for parsers: local and global ambiguities, misspellings, ill-formed grammar, non-sequitur, conversation, deeply embedded clauses, technical jargon, and parentheticals. The overall theme of the present inquiry was to see whether the parsers somehow capture semantic information and translate that information into augmenting structural analyses of the given statements. The orientation of the present approach was from that of the human language user: it was assumed that the human use of language is the gold standard to which machine-based language parsing should be held.

Sentences used

1. Flying planes made her duck.
2. They read with me.
3. Dark shining ghosts vastly drink cities.
4. John is going probably to home.
5. i drov home frm th licor stor.
6. And so Billy, he, like, he totally didn't, you know?
7. You don't say.
8. Government is number.
9. Short-lived like a machine that is used but not good enough whilst promising to be better, an enduring work must be built like a machine full of shortcomings.
10. During the automatic customization (or training) of MSR-MT (see figure below), pairs of corresponding source and target sentences are parsed to produce graph-like structures called Logical Forms (LFs).

Parsers used

Memory-based shallow parser demo:
http://ilk.kub.nl/cgi-bin/tstchunk/demo.pl

EP41R Parser
http://www.cs.kun.nl/agfl/ep4ir/try.html

Connexor machinese for English (syntax tree output)
http://www.connexor.com/demos/syntax_en.html

Selection Justification

1. *Flying planes made her duck.*

"This example is a globally ambiguous sentence; that is, the entire string of words has more than one structure associated with it."[1]

I plucked the above example because it appears to be a fine example of the way one sentence can have complexes of meaning. Context is the only way to possibly pick one possibility over another. The present example is a difficult sentence for a parser to handle.

I find there are at least six central ways of interpreting the above in a 3 x 2 matrix:

Flying planes	*Made her duck*
Her flying of a plane	manufactured her rubber duckie.
Her flying in a plane	encouraged her to squat.
Planes that are flying above	

1. Her flying of a plane manufactured her rubber duckie.
2. Her flying of a plane encouraged her to squat.
3. Her flying in a plane manufactured her rubber duckie.
4. Her flying in a plane encouraged her to squat.
5. Planes that are flying above manufactured her rubber duckie.
6. Planes that are flying above encouraged her to squat.

Many other possibilities can be suggested though they tend towards more and more remotely possible worlds. It seems that only 6 seems likely, while 4 and 5 are only tenuous possibilities. We therefore want to see a parsing that favors 6, and maybe tries to pass off 4 or 5 instead.

We want to see something like:
[NP [ADJ Flying] [N planes][VP [V made] [OBJ her] [V INF duck]

where the "duck" is treated as the infinitive form, as in "Judy made Bill cry."

Will these parsers do a good job resolving the ambiguities of constituency?

---

[1] in *Natural Language Processing in Prolog/Pop11/Lisp*, Gerald Gazdar & Chris Mellish; see http://www.informatics.susx.ac.uk/research/nlp/gazdar/nlp-in-prolog/ch01/chapter-01-sh-1.2.html

2. *They read with me.*

"Read" is ambiguous with respect to tense (is it present or past tense?). The global ambiguity example was a difficult test, and I expect none of the tools will have fared well with it. The present example contains a simple word-level ambiguity and should be at least easier to handle.

3. *Dark shining ghosts vastly drink cities.*

The present example tests a parser's ability to handle well-formed structures containing nonsense words—ultimately a syntactically correct but semantically troublesome sentence. The present example was inspired by Chomsky's famous example, "colorless green ideas sleep furiously." I would have used Chomsky's own example but I feared it might risk an encounter with a "special case" written into the parser.

4. *Sally is going probably to home.*

With this example I wanted to choose an obviously ill-formed sentence, one that humans (or, rather, perhaps only English speakers) simply cannot comprehend. In this case we can understand it, but it reads so awkwardly that it is barely comprehendible.

What is an ill-formed sentence? "Much 'naturally occurring' text contains some or many typographical errors or other errors. Industrial-strength parsers have to be able to deal with these, just as people can deal with typos and ungrammaticality. Such a parser is called a robust parser."[2]

5 *i drov home frm th licor stor*.

I devised the following statement because of two overlapping linguistic phenomena that do not prevent human comprehension: letter elision and misspelling. We tend to be able to understand words that are missing their first or last letters, or are missing their vowels. Also words that are spelled in a way more "true" to their phonetic spelling tend to be understandable. We know that the above statement is the rather disturbing, given the spelling, "I drove home from the liquor store."

6. *And so Billy, he goes, like, he was like, he totally didn't, you know?*

While at lunch one day two weeks ago I overheard some students engaged in an energetic conversation about their social lives. I was and continue to be amazed with the facility

[2] from The Natural Language Processing Dictionary, Bill Wilson, 2004, http://www.cse.unsw.edu.au/~billw/nlpdict.html#ill-formed

many a young American has with the words “like,” “totally,” “goes,” and the phrase “you know.” We do not see such statements frequently in text though at some point, with the growth of speech recognition software, we may see such text more often and may need to be able to parse it. The crux of this statement is expressive (in Searle’s sense) and so tests the facility of the parser to handle not only speech acts but also recapitulations: note that the verb is revised by the speaker twice.

7. *You don't say.*

Another expressive, but a much simpler example than the previous. Here, we have a transitive verb that is treated as if it is intransitive. It computes to the human listener, yet it may fall outside the margins of what a typical NLP system might successfully parse.

8. *Government is number.*

I have no idea what it means. Maybe a parser might help me.

I picked the present example because it seems strange (inspired by something presented earlier this semester from our guests from DAS in reference to a Sergei Brin paper:

> We include the threesome "government," "is," and "number" because it has the highest value of any triple of words. Like many of the correlated triples, of which there are well over a million, this itemset is hard to interpret. Part of the difficulty is due to the word "is," which does not yield as much context as nouns and active verbs. In practice, it may make sense to restrict the analysis to nouns and active verbs to prune away such meaningless correlates.[3]

Specifically, the present example is a remnant not of human text but of machine text. I wonder whether machine-generated text poses difficulties for natural language parsers.

9. *Short-lived like a machine that is used but not good enough whilst promising to be better, an enduring work must be built like a machine full of shortcomings.*

> - Bertolt Brecht, from "About the Way to Construct Enduring Works" (my translation, though I wouldn't typically use 'whilst')

It's a strange phrasing, in English as it was in German, but in English it has the dramatic oddity of Germanic verb latency (verbs in German tend to appear towards the end of sentences) and with modification of the main part of the sentence through a phrase

---

[3] Sergey Brin, Rajeev Motwani, Craig Silverstein, “Beyond Market Baskets: Generalizing Association Rules to Correlations.” SIGMOD 1997, Proceedings ACM SIGMOD International Conference on Management of Data, May 13-15, 1997, Tucson, Arizona, USA

repeating "like". Also, here's an opportunity to test an old word like “whilst.” The present example also contains deeply embedded phrases: “short lived” modifies “an enduring work”, and “short lived” is modified by a long subordinate clause with its own subordinates.

10. *During the automatic customization (or training) of MSR-MT (see figure below), pairs of corresponding source and target sentences are parsed to produce graph-like structures called Logical Forms (LFs).*

This example is a grab-bag of sorts: a long complex sentence with different types of parentheticals, with technical content, with ostensive information (a speech act) and with meta-content (*i.e.*, the sentence pertains to NLP).

Many of the above examples are quite difficult, but their difficulty will help in the present effort to articulate the shortcomings of NLP parsers and illuminate opportunities for improvement.

Results & Analysis

1. *Flying planes made her duck.*

Memory-based shallow parser demo:

Tagger output

Flying/NN planes/NNS made/VBD her/PRP$ duck/NN ./.

Chunker output

[NP Flying/NN planes/NNS NP] [VP made/VBD VP] [NP her/PRP$ duck/NN NP] ./.

Subject/Object Detector output

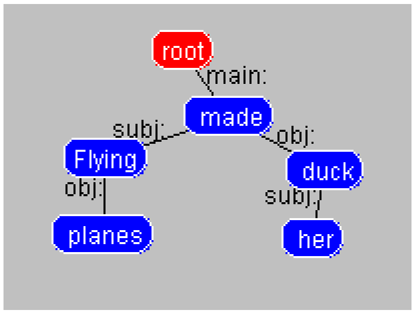
[NP_1NP-SBJ;NP-OBJ Flying/NN planes/NNS NP_1NP-SBJ;NP-OBJ] [VP_1 made/VBD VP_1] [NP_1^{Object} her/PRP$ duck/NN NP_1^{Object}] ./.

EP41R Parser:

Parse tree for the most probable analysis:

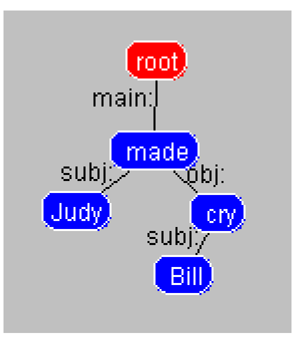
```
segment
  sentence
    statement
      simple statement
        SVOC phrase
          opt circumstances
          subject(sing, third)
            NP(sing, third, nom)
              noun phrase(sing, third, nom)
                noun part(sing, third, nom)
                  noun group(sing, third, nom)
                    noun kernel(sing)
                      LEX_NOUN(sing)
                        LNOUN(sing)
                          NOUN(sing)
                            "Flying"
          VOC phrase(sing, third)
            verb group(trav, none, none, sing, third)
              verb form(trav, none, none, sing, third)
                LEX_VERBS(none, trav)
                  VERBS(none, trav)
                    "planes"
              pred adjectives
                pred adjective
                  adjective
                    LEX_ADJE(abso)
                      LADJE(abso)
                        ADJE(abso)
                          "made"
            OC phrase(trav, none, none)
              object
                noun phrase(sing, third, acc)
                  noun part(sing, third, acc)
                    poss pron
                      LEX_POSSPRON
                        POSSPRON
                          "her"
                    noun part(sing, third, acc)
                      noun group(sing, third, acc)
                        noun kernel(sing)
                          LEX_NOUN(sing)
                            LNOUN(sing)
                              NOUN(sing)
                                "duck"
              particle(none)
              C phrase(none)
                opt circumstances
    period
      "."
```

Connexor machinese for English (syntax tree output):



#	Text	Baseform	Syntactic relation	Syntax and morphology
1	Flying	fly	subj:>3	@-FMAINV %VA ING
2	planes	plane	obj:>1	@OBJ %NH N NOM PL
3	made	make	main:>0	@+FMAINV %VA V PAST
4	her	she	subj:>5	@OBJ %NH PRON PERS ACC SG3
5	duck	duck	obj:>3	@-FMAINV %VA V INF
6	.	.		
7	<s>	<s>		

Compare with “Judy made Bill cry.”



#	Text	Baseform	Syntactic relation	Syntax and morphology
1	Judy	judy	subj:>2	@SUBJ %NH N NOM SG
2	made	make	main:>0	@+FMAINV %VA V PAST
3	Bill	bill	subj:>4	@OBJ %NH N NOM SG
4	cry	cry	obj:>2	@-FMAINV %VA V INF
5	.	.		
6	<s>	<s>		

2. *They read with me.*

Memory-based shallow parser demo:

Tagger output

They/PRP read/VBP with/IN me/PRP ./.

Chunker output

[NP They/PRP NP] [VP read/VBP VP] {PNP [Prep with/IN Prep] [NP me/PRP NP] PNP} ./.

Subject/Object Detector output

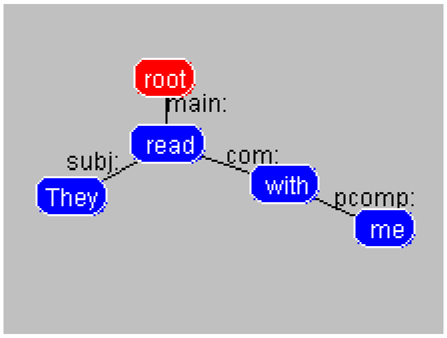
[$NP_1^{Subject}$ They/PRP $NP_1^{Subject}$] [VP_1 read/VBP VP_1] {PNP [P with/IN P] [NP me/PRP NP] PNP} ./.

EP41R Parser:

Parse tree for the most probable analysis:

```
segment
  sentence
    statement
      simple statement
        SVOC phrase
          opt circumstances
          subject(plur, third)
            NP(plur, third, nom)
              noun phrase(plur, third, nom)
                noun part(plur, third, nom)
                  noun group(plur, third, nom)
                    pers pron(plur, third, nom)
                      LEX_PERSPRON(plur, third, nom)
                        PERSPRON(plur, third, nom)
                          "they"
          VOC phrase(plur, third)
            verb group(trav, none, from|into|in, plur, third)
              verb form(trav, none, from|into|in, plur, third)
                LEX_VERBI(from|into|in, trav)
                  VERBI(from|into|in, trav)
                    "read"
            OC phrase(trav, none, from|into|in)
              particle(none)
              C phrase(from|into|in)
                instrument
                  PP(with)
                    opt adverbs
                    LEX_PREPOS(with)
                      PREPOS(with)
                        "with"
                    noun phrase(sing, first, dat)
                      noun part(sing, first, dat)
                        noun group(sing, first, dat)
                          pers pron(sing, first, dat)
                            LEX_PERSPRON(sing, first, dat)
                              PERSPRON(sing, first, dat)
                                "me"
                C phrase(from|into|in)
                  opt circumstances
    period
      "."
```

Connexor machinese for English (syntax tree output):



#	Text	Baseform	Syntactic relation	Syntax and morphology
1	They	they	subj:>2	@SUBJ %NH PRON PERS NOM PL3
2	read	read	main:>0	@+FMAINV %VA V PAST
3	with	with	com:>2	@ADVL %EH PREP
4	me	i	pcomp:>3	@<P %NH PRON PERS ACC SG1
5	.	.		
6	<s>	<s>		

3. *Dark shining ghosts vastly drink cities.*

Memory-based shallow parser demo:

Tagger output

Dark/JJ shining/VBG ghosts/NNS vastly drink/VBP cities/NNS ./.

Chunker output

[NP Dark/JJ shining/VBG ghosts/NNS NP] vastly/RB
[VP drink/VBP VP] [NP cities/NNS NP] ./.

Subject/Object Detector output

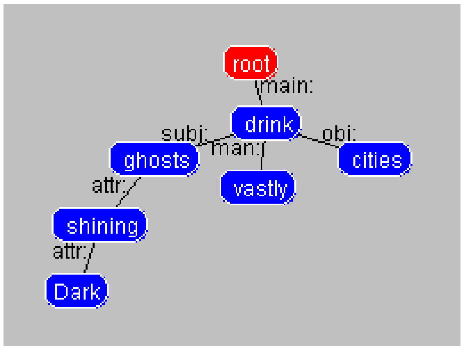
[$NP_1^{Subject}$ Dark/JJ shining/VBG ghosts/NNS $NP_1^{Subject}$] [ADVP vastly/RB ADVP] [VP_1 drink/VBP VP_1] [NP_1^{Object} cities/NNS NP_1^{Object}] ./.

EP41R Parser:

Parse tree for the most probable analysis:

```
segment
  sentence
    statement
      simple statement
        SVOC phrase
          opt circumstances
            circumstance
              adjective
                LEX_ADJE(abso)
                  LADJE(abso)
                    ADJE(abso)
                      "dark"
            opt circumstances
          subject(sing, third)
            NP(sing, third, nom)
              verbal noun phrase
                opt adverbs
                pOC phrase(prpl)
                  participle(intr, none, none, prpl)
                    verb form(intr, none, none, prpl)
                      LEX_VERBG(none, intr)
                        VERBG(none, intr)
                          "shining"
                  OC phrase(intr, none, none)
                    C phrase(none)
                      opt circumstances
          VOC phrase(sing, third)
            verb group(trav, none, none, sing, third)
              verb form(trav, none, none, sing, third)
                LEX_VERBS(none, trav)
                  VERBS(none, trav)
                    "ghosts"
            OC phrase(trav, none, none)
              object
                noun phrase(plur, third, acc)
                  adverb
                    LEX_ADVB
                      LADVB
                        ADVB
                          "vastly"
                  noun phrase(plur, third, acc)
                    noun part(plur, third, acc)
                      noun group(plur, third, acc)
                        noun kernel(plur)
                          premodifiers
                            premodifier
                              LEX_NOUN(sing)
                                LNOUN(sing)
                                  NOUN(sing)
                                    "drink"
                            rest premodifiers
                          noun kernel(plur)
                            LEX_NOUN(plur)
                              LNOUN(plur)
                                NOUN(plur)
                                  "cities"
              particle(none)
              C phrase(none)
                opt circumstances
    period
      "."
```

Connexor machinese for English (syntax tree output)



#	Text	Baseform	Syntactic relation	Syntax and morphology
1	Dark	dark	attr:>2	@A> %>N A ABS
2	shining	shining	attr:>3	@A> %>N A ABS
3	ghosts	ghost	subj:>5	@SUBJ %NH N NOM PL
4	vastly	vastly	man:>5	@ADVL %EH ADV
5	drink	drink	main:>0	@+FMAINV %VA V PRES
6	cities	city	obj:>5	@OBJ %NH N NOM PL
7	.	.		
8	<s>	<s>		

4. *Sally is going probably to home.*

Memory-based shallow parser demo:

Tagger output

Sally/NN is/VBZ going/VBG probably to/TO home/VB ./.

Chunker output

[NP Sally/NN NP] [VP is/VBZ going/VBG probably/RB to/TO home/VB VP] ./.

Subject/Object Detector output

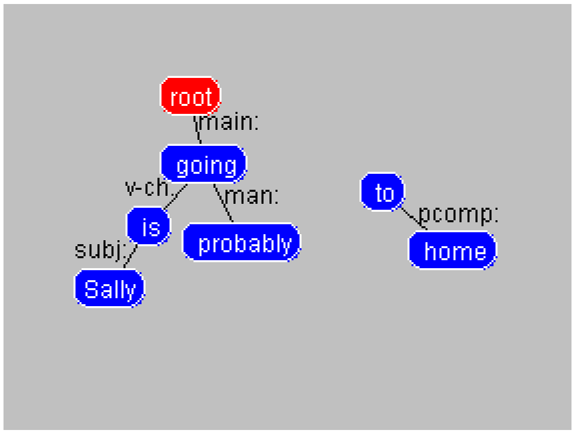
[NP$_1^{Subject}$ Sally/NN NP$_1^{Subject}$] [VP$_1$ is/VBZ going/VBG probably/RB to/TO home/VB VP$_1$] ./.

EP41R Parser

Parse tree for the most probable analysis:

```
segment
  sentence
    statement
      simple statement
        SVOC phrase
          opt circumstances
          subject(sing, third)
            NP(sing, third, nom)
              noun phrase(sing, third, nom)
                noun part(sing, third, nom)
                  noun group(sing, third, nom)
                    noun kernel(sing)
                      LEX_NOUN(sing)
                        LNOUN(sing)
                          NOUN(sing)
                            "sally"
          VOC phrase(sing, third)
            verb group(intr, none, to, sing, third)
              LEX_TOBE(sing, third)
                TOBE(sing, third)
                  "is"
              opt adverbs
              participle(intr, none, to, prpl)
                verb form(intr, none, to, prpl)
                  LEX_VERBG(to, intr)
                    VERBG(to, intr)
                      "going"
            OC phrase(intr, none, to)
              C phrase(to)
                PP(to)
                  opt adverbs
                    adverb
                      LEX_ADVB
                        LADVB
                          ADVB
                            "probably"
                    opt adverbs
                  LEX_PREPOS(to)
                    PREPOS(to)
                      "to"
                  noun phrase(sing, third, dat)
                    noun part(sing, third, dat)
                      noun group(sing, third, dat)
                        noun kernel(sing)
                          LEX_NOUN(sing)
                            LNOUN(sing)
                              NOUN(sing)
                                "home"
                opt circumstances
    period
      "."
```

Connexor machinese for English (syntax tree output)



#	Text	Baseform	Syntactic relation	Syntax and morphology
1	Sally	sally	subj:>2	@SUBJ %NH N NOM SG
2	is	be	v-ch:>3	@+FAUXV %AUX V PRES SG3
3	going	go	main:>0	@-FMAINV %VA ING
4	probably	probably	man:>3	@ADVL %EH ADV
5	to	to		@ADVL %EH PREP
6	home	home	pcomp:>5	@<P %NH N NOM SG
7	.	.		
8	<s>	<s>		

5 *i drov home frm th licor stor.*

Memory-based shallow parser demo:

Tagger output

i//FW drov//NN home/NN frm//NN th//NN licor//NN stor//NN ./.

Chunker output

[NP i//FW drov//NN home/NN frm//NN th//NN licor//NN stor//NN NP] ./.

Subject/Object Detector output

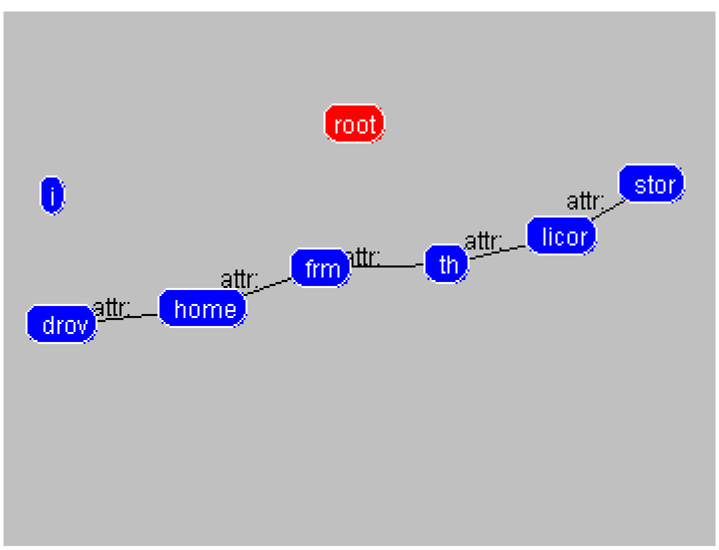
[NP i//FW drov//NN home/NN frm//NN th//NN licor//NN stor//NN NP] ./.

EP41R Parser

Parse tree for the most probable analysis:

```
Warning: skipped unknown token "i" at pos 0

Warning: skipped unknown token "drov" at pos 2

Warning: skipped unknown token "frm" at pos 12

Warning: skipped unknown token "th" at pos 16

Warning: skipped unknown token "licor" at pos 19

Warning: skipped unknown token "stor" at pos 25


segment
  sentence
    order
      VOC phrase(sing, secnd)
        verb group(trav, none, on, sing, secnd)
          verb form(trav, none, on, sing, secnd)
            LEX_VERBI(on, trav)
              VERBI(on, trav)
                "home"
        OC phrase(trav, none, on)
          particle(none)
          C phrase(on)
            opt circumstances
```

Connexor machinese for English (syntax tree output)



#	Text	Baseform	Syntactic relation	Syntax and morphology
1	i	i		@SUBJ %NH PRON PERS NOM SG1
2	drov	drov	attr:>3	@A> %>N <?> N NOM SG
3	home	home	attr:>4	@A> %>N N NOM SG
4	frm	frm	attr:>5	@A> %>N <?> N NOM SG
5	th	th	attr:>6	@A> %>N ABBR NOM SG
6	licor	licor	attr:>7	@A> %>N <?> N NOM SG
7	stor	stor		@NH %NH <?> N NOM SG
8	.	.		
9	<s>	<s>		

6. *And so Billy, he goes, like, he was like, he totally didn't, you know?*

Memory-based shallow parser demo:

Tagger output

And/CC so Billy/NNP ,/, he/PRP goes/VBZ ,/, like/IN ,/, he/PRP was/VBD like/IN ,/, he/PRP totally did/VBD n't ,/, you/PRP know/VB ?/.

Chunker output

And/CC so/RB [NP Billy/NNP NP] ,/, [NP he/PRP NP] [VP goes/VBZ VP] ,/, [Prep like/IN Prep] ,/, [NP he/PRP NP] [VP was/VBD VP] [Prep like/IN Prep] ,/, [NP he/PRP NP] [ADJP totally/RB ADJP] [VP did/VBD VP] n't/RB ,/, [NP you/PRP NP] [VP know/VB VP] ?/.

Subject/Object Detector output

And/CC [ADVP so/RB ADVP] [NP Billy/NNP NP] ,/, [$NP_1^{Subject}$ he/PRP $NP_1^{Subject}$] [VP_1 goes/VBZ VP_1] ,/, [P like/IN P] ,/, [$NP_2^{Subject}$ he/PRP $NP_2^{Subject}$] [VP_2 was/VBD VP_2] [P like/IN P] ,/, [$NP_3^{Subject}$ he/PRP $NP_3^{Subject}$] [ADJP totally/RB ADJP] [VP_3 did/VBD VP_3] n't/RB ,/, [$NP_4^{Subject}$ you/PRP $NP_4^{Subject}$] [VP_4 know/VB VP_4] ?/.

EP41R Parser:

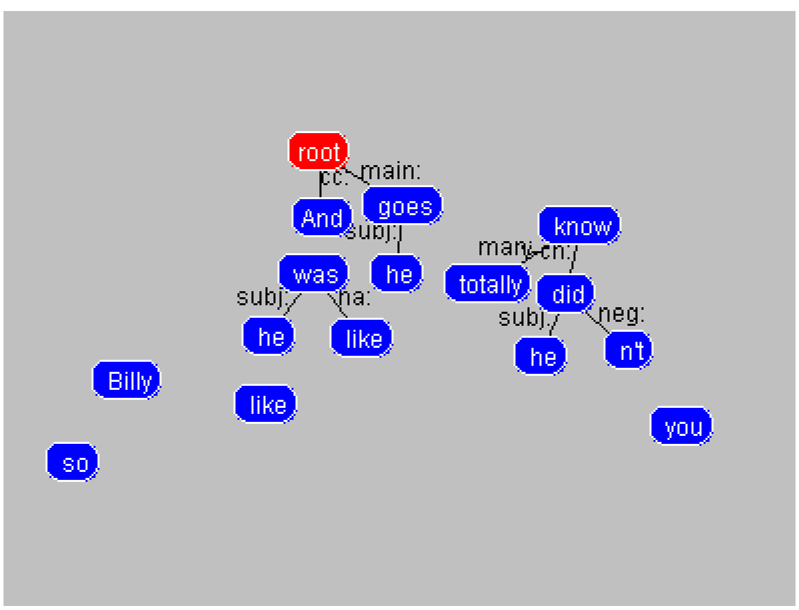
Parse tree for the most probable analysis:

```
Warning: skipped unknown token "'t" at pos 57


segment
  NP
    noun phrase(plur, third, nom)
      adverb
        LEX_ADVB
          LADVB
            ADVB
              "and so"
      noun phrase(plur, third, nom)
        noun part(sing, third, nom)
          noun group(sing, third, nom)
            noun kernel(sing)
              LEX_NOUN(sing)
                LNOUN(sing)
                  NOUN(sing)
                    "billy"
        coordinator
          LEX_CON(coo)
            CON(coo)
              ","
        noun phrase(sing, third, nom)
          noun part(sing, third, nom)
            noun group(sing, third, nom)
              pers pron(sing, third, nom)
                LEX_PERSPRON(sing, third, nom)
                  PERSPRON(sing, third, nom)
                    "he"


total number of parsings 3 (max 1)
total scan time 0.014
total parse time 0.003


total number of parsings 0 (max 1)
total scan time 0.014
total parse time 0.003


total number of parsings 0 (max 1)
total scan time 0.014
total parse time 0.004

segment
  sentence
    order
      VOC phrase(sing, secnd)
        verb group(trav, none, none, sing, secnd)
          verb form(trav, none, none, sing, secnd)
            LEX_VERBI(none, trav)
              VERBI(none, trav)
                "like"
        OC phrase(trav, none, none)
          particle(none)
          C phrase(none)
            opt circumstances


total number of parsings 1 (max 1)
total scan time 0.014
total parse time 0.004


total number of parsings 0 (max 1)
```

```
total scan time 0.014
total parse time 0.004

segment
  NP
    noun phrase(sing, third, nom)
      noun part(sing, third, nom)
        noun group(sing, third, nom)
          pers pron(sing, third, nom)
            LEX_PERSPRON(sing, third, nom)
              PERSPRON(sing, third, nom)
                "he"


total number of parsings 1 (max 1)
total scan time 0.014
total parse time 0.005


total number of parsings 0 (max 1)
total scan time 0.014
total parse time 0.005

segment
  sentence
    order
      VOC phrase(sing, secnd)
        verb group(trav, none, none, sing, secnd)
          verb form(trav, none, none, sing, secnd)
            LEX_VERBI(none, trav)
              VERBI(none, trav)
                "like"
        OC phrase(trav, none, none)
          particle(none)
          C phrase(none)
            opt circumstances


total number of parsings 1 (max 1)
total scan time 0.014
total parse time 0.005


total number of parsings 0 (max 1)
total scan time 0.014
total parse time 0.005

segment
  NP
    noun phrase(sing, third, nom)
      noun part(sing, third, nom)
        noun group(sing, third, nom)
          pers pron(sing, third, nom)
            LEX_PERSPRON(sing, third, nom)
              PERSPRON(sing, third, nom)
                "he"
          rel phrase(acc)
            subject(plur, third)
              NP(plur, third, nom)
                noun phrase(plur, third, nom)
                  adverb
                    LEX_ADVB
                      LADVB
                        ADVB
                          "totally"
                  noun phrase(plur, third, nom)
                    noun part(plur, third, nom)
                      noun group(plur, third, nom)
                        noun kernel(plur)
                          LEX_NOUN(plur)
                            LNOUN(plur)
```

```
                              NOUN(plur)
                                "didn"
                    coordinator
                      LEX_CON(coo)
                        CON(coo)
                          ","
                    noun phrase(NUMB, secnd, nom)
                      noun part(NUMB, secnd, nom)
                        noun group(NUMB, secnd, nom)
                          pers pron(NUMB, secnd, nom)
                            LEX_PERSPRON(NUMB, secnd, nom)
                              PERSPRON(NUMB, secnd, nom)
                                "you"
            verb group(trav, none, none, plur, third)
              verb form(trav, none, none, plur, third)
                LEX_VERBI(none, trav)
                  VERBI(none, trav)
                    "know"
            pref PP(none)
            opt circumstances


total number of parsings 2 (max 1)
total scan time 0.014
total parse time 0.006


total number of parsings 0 (max 1)
total scan time 0.014
total parse time 0.006


total number of parsings 0 (max 1)
total scan time 0.014
total parse time 0.006
```

Connexor machinese for English (syntax tree output)



#	Text	Baseform	Syntactic relation	Syntax and morphology
1	And	and	cc:>0	@CC %CC CC
2	so	so		@ADVL %EH ADV
				@AD-A> %E> ADV
3	Billy	billy		@NH %NH N NOM SG
				@PCOMPL-S %NH N NOM SG
				@OBJ %NH N NOM SG
				@SUBJ %NH N NOM SG
4	,	,		
5	he	he	subj:>6	@SUBJ %NH PRON PERS NOM SG3
6	goes	go	main:>0	@+FMAINV %VA V PRES SG3
7	,	,		
8	like	like		@+FMAINV %VA V PRES
9	,	,		
10	he	he	subj:>11	@SUBJ %NH PRON PERS NOM SG3
11	was	be		@+FMAINV %VA V PAST
12	like	like	ha:>11	@ADVL %EH PREP
13	,	,		
14	he	he	subj:>16	@SUBJ %NH PRON PERS NOM SG3
15	totally	totally	man:>20	@ADVL %EH ADV
16	did	do	v-ch:>20	@+FAUXV %AUX V PAST
17	n't	not	neg:>16	@ADVL %EH NEG-PART
18	,	,		
19	you	you		@SUBJ %NH PRON PERS NOM
				@<P %NH PRON PERS NOM
				@PCOMPL-S %NH PRON PERS NOM
20	know	know		@-FMAINV %VA V INF
21	?	?		
22	<p>	<p>		

7. *You don't say.*

Memory-based shallow parser demo:

Tagger output

You/PRP do/VBP n't say/VB ./.

Chunker output

[NP You/PRP NP] [VP do/VBP n't/RB say/VB VP] ./.

Subject/Object Detector output

[$NP_1^{Subject}$ You/PRP $NP_1^{Subject}$] [VP_1 do/VBP n't/RB say/VB VP_1] ./.

EP41R Parser:

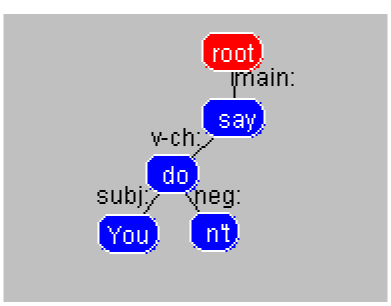
Parse tree for the most probable analysis:

```
Warning: skipped unknown token "'t" at pos 7


segment
  sentence
    statement
      simple statement
        SVOC phrase
          opt circumstances
          subject(NUMB, secnd)
            NP(NUMB, secnd, nom)
              noun phrase(NUMB, secnd, nom)
                noun part(NUMB, secnd, nom)
                  noun group(NUMB, secnd, nom)
                    pers pron(NUMB, secnd, nom)
                      LEX_PERSPRON(NUMB, secnd, nom)
                        PERSPRON(NUMB, secnd, nom)
                          "you"
          VOC phrase(NUMB, secnd)
            verb group(trav, none, to, NUMB, secnd)
              LEX_AUXV(NUMB, secnd)
                AUXV(NUMB, secnd)
                  "don't"
              infinitive(trav, none, to)
                verb form(trav, none, to, infi)
                  LEX_VERBI(to, trav)
                    VERBI(to, trav)
                      "say"
            OC phrase(trav, none, to)
              particle(none)
              C phrase(to)
                opt circumstances
    period
      "."
```

Compare with “You do not say”:

Parse tree for the most probable analysis:
```
segment
  sentence
    statement
      simple statement
        SVOC phrase
          opt circumstances
          subject(NUMB, secnd)
            NP(NUMB, secnd, nom)
              noun phrase(NUMB, secnd, nom)
                noun part(NUMB, secnd, nom)
                  noun group(NUMB, secnd, nom)
                    pers pron(NUMB, secnd, nom)
                      LEX_PERSPRON(NUMB, secnd, nom)
                        PERSPRON(NUMB, secnd, nom)
                          "you"
          VOC phrase(NUMB, secnd)
            verb group(trav, none, to, NUMB, secnd)
              LEX_AUXV(NUMB, secnd)
                AUXV(NUMB, secnd)
                  "do"
              infinitive(trav, none, to)
                adverb
                  LEX_ADVB
                    LADVB
                      ADVB
                        "not"
                infinitive(trav, none, to)
                  verb form(trav, none, to, infi)
                    LEX_VERBI(to, trav)
                      VERBI(to, trav)
                        "say"
            OC phrase(trav, none, to)
              particle(none)
              C phrase(to)
                opt circumstances
    period
      "."
```

Connexor machinese for English (syntax tree output)



#	Text	Baseform	Syntactic relation	Syntax and morphology
1	You	you	subj:>2	@SUBJ %NH PRON PERS NOM
2	do	do	v-ch:>4	@+FAUXV %AUX V PRES
3	n't	not	neg:>2	@ADVL %EH NEG-PART
4	say	say	main:>0	@-FMAINV %VA V INF
5	.	.		
6	<s>	<s>		

8. *Government is number.*

Memory-based shallow parser demo:

Tagger output

Government/NN is/VBZ number/NN ./.

Chunker output

[NP Government/NN NP] [VP is/VBZ VP] [NP number/NN NP] ./.

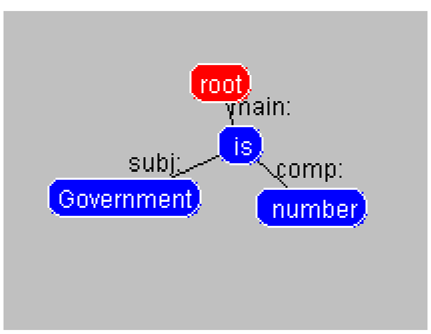
Subject/Object Detector output

[$NP_1^{Subject}$ Government/NN $NP_1^{Subject}$] [VP_1 is/VBZ VP_1] [NP_1NP-PRD number/NN NP_1NP-PRD] ./.

EP41R Parser:

Parse tree for the most probable analysis:
```
segment
  sentence
    statement
      simple statement
        SVOC phrase
          opt circumstances
          subject(sing, third)
            NP(sing, third, nom)
              noun phrase(sing, third, nom)
                noun part(sing, third, nom)
                  noun group(sing, third, nom)
                    noun kernel(sing)
                      LEX_NOUN(sing)
                        LNOUN(sing)
                          NOUN(sing)
                            "government"
          xP phrase(sing, third)
            copula(sing, third)
              to be(sing, third)
                LEX_TOBE(sing, third)
                  TOBE(sing, third)
                    "is"
            predicate
              noun phrase(sing, third, nom|acc)
                noun part(sing, third, nom|acc)
                  noun group(sing, third, nom|acc)
                    noun kernel(sing)
                      LEX_NOUN(sing)
                        LNOUN(sing)
                          NOUN(sing)
                            "number"
    period
      "."
```

Connexor machinese for English (syntax tree output)



#	Text	Baseform	Syntactic relation	Syntax and morphology
1	Government	government	subj:>2	@SUBJ %NH N NOM
2	is	be	main:>0	@+FMAINV %VA V PRES SG3
3	number	number	comp:>2	@PCOMPL-S %NH N NOM
4	.	.		
5	<s>	<s>		

9. *Short-lived like a machine that is used but not good enough whilst promising to be better, an enduring work must be built like a machine full of shortcomings.*

Memory-based shallow parser demo:

Tagger output

Short-lived//JJ like/IN a/DT machine/NN that/WDT is/VBZ used/VBN but/CC not good/JJ enough whilst//JJS promising/JJ to/TO be/VB better ,/, an/DT enduring/VBG work/NN must/MD be/VB built/VBN like/IN a/DT machine/NN full/JJ of/IN shortcomings/NNS ./.

Chunker output

[ADJP Short-lived//JJ ADJP] {PNP [Prep like/IN Prep] [NP a/DT machine/NN NP] PNP} [NP that/WDT NP] [VP is/VBZ used/VBN VP] but/CC not/RB [ADJP good/JJ enough/RB ADJP] [NP whilst//JJS promising/JJ NP] [VP to/TO be/VB VP] better/RB ,/, [NP an/DT enduring/VBG work/NN NP] [VP must/MD be/VB built/VBN VP] {PNP [Prep like/IN Prep] [NP a/DT machine/NN NP] PNP} [ADJP full/JJ ADJP] {PNP [Prep of/IN Prep] [NP shortcomings/NNS NP] PNP} ./.

Subject/Object Detector output

[ADJP Short-lived//JJ ADJP] {PNP [P like/IN P] [NP a/DT machine/NN NP] PNP} [NP$_1^{Subject}$ that/WDT NP$_1^{Subject}$] [VP$_1$ is/VBZ used/VBN VP$_1$] but/CC not/RB [ADJP good/JJ enough/RB ADJP] [NP$_2^{Subject}$ whilst//JJS promising/JJ NP$_2^{Subject}$] [VP$_2$ to/TO be/VB VP$_2$] [ADVP better/RB ADVP] ,/, [NP$_3^{Subject}$ an/DT enduring/VBG work/NN NP$_3^{Subject}$] [VP$_3$ must/MD be/VB built/VBN VP$_3$] {PNP [P like/IN P] [NP a/DT machine/NN NP] PNP} [ADJP full/JJ ADJP] {PNP [P of/IN P] [NP shortcomings/NNS NP] PNP} ./.

EP41R Parser:

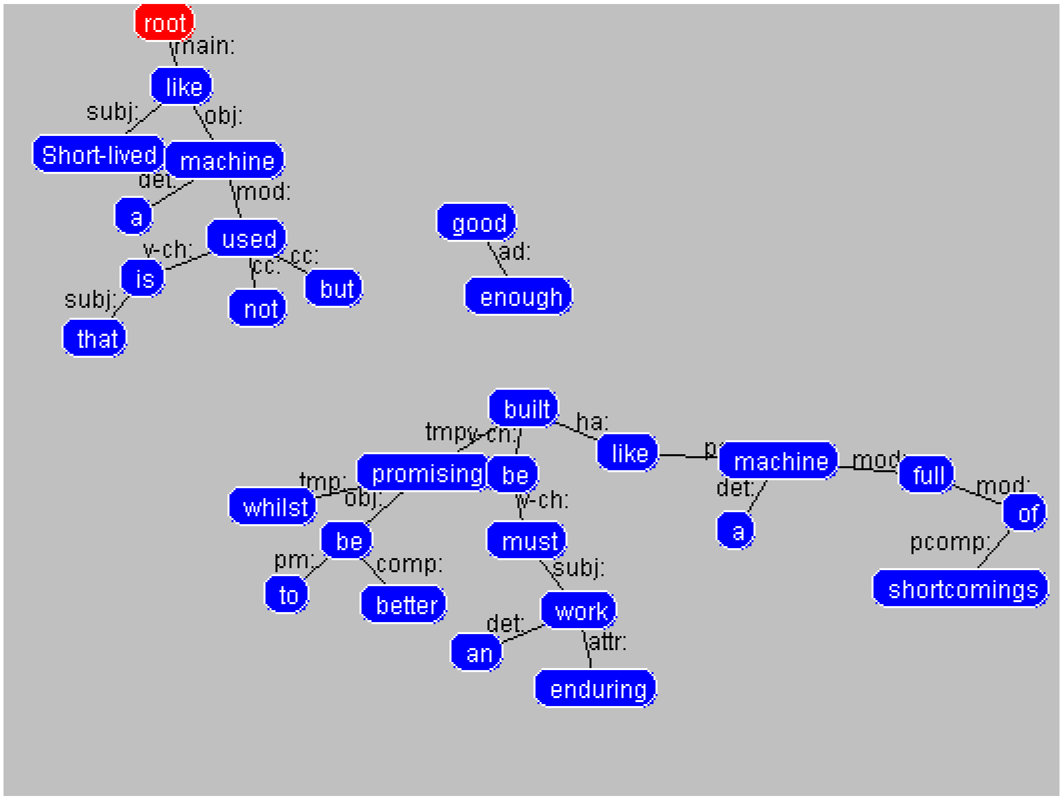
Parse tree for the most probable analysis:

```
segment
  sentence
    statement
      simple statement
        SVOC phrase
          opt circumstances
            circumstance
              adjective
                LEX_ADJE(abso)
                  LADJE(abso)
                    ADJE(abso)
                      "short-lived"
            opt circumstances
              circumstance
                PP
                  PP(like)
                    opt adverbs
                    LEX_PREPOS(like)
                      PREPOS(like)
                        "like"
                    noun phrase(sing, third, dat)
                      noun part(sing, third, dat)
                        article(sing)
                          LEX_ART(sing)
                            ART(sing)
                              "a"
                        noun group(sing, third, dat)
                          noun kernel(sing)
                            LEX_NOUN(sing)
                              LNOUN(sing)
                                NOUN(sing)
                                  "machine"
              opt circumstances
          subject(sing, third)
            NP(sing, third, nom)
              noun phrase(sing, third, nom)
                noun part(sing, third, nom)
                  noun group(sing, third, nom)
                    pers pron(sing, third, nom)
                      LEX_PERSPRON(sing, third, nom)
                        PERSPRON(sing, third, nom)
                          "that"
          xP phrase(sing, third)
            copula(sing, third)
              to be(sing, third)
                LEX_TOBE(sing, third)
                  TOBE(sing, third)
                    "is"
            predicate
              pred adjectives
                pred adjective
                  adjective
                    LEX_ADJE(abso)
                      LADJE(abso)
                        ADJE(abso, to)
                          "used"
                coordinator
                  LEX_CON(coo)
                    CON(coo)
                      "but"
                pred adjectives
                  pred adjective
                    adjective
                      adverb
                        LEX_ADVB
                          LADVB
                            ADVB
```

```
                "not"
            adjective
              LEX_ADJE(abso)
                LADJE(abso)
                  ADJE(abso)
                    "good"
    opt circumstances
      circumstance
        adverb
          LEX_ADVB
            LADVB
              ADVB
                "enough"
      opt circumstances
        circumstance
          subordinator
            LEX_CON(sub)
              CON(sub)
                "whilst"
          simple statement
            SVOC phrase
              opt circumstances
                circumstance
                  adjective
                    LEX_ADJE(abso)
                      LADJE(abso)
                        ADJE(abso)
                          "promising"
                opt circumstances
                  circumstance
                    purpose
                      "to"
                      LEX_TOBE(infi)
                        TOBE(infi)
                          "be"
                      predicate
                        pred adjectives
                          pred adjective
                            adjective
                              LEX_ADJE(comp)
                                LADJE(comp)
                                  ADJE(comp)
                                    "better"
                        opt circumstances
                  comma
                    ","
                  opt circumstances
              subject(sing, third)
                NP(sing, third, nom)
                  noun phrase(sing, third, nom)
                    noun part(sing, third, nom)
                      article(sing)
                        LEX_ART(sing)
                          ART(sing)
                            "an"
                      noun group(sing, third, nom)
                        noun kernel(sing)
                          premodifiers
                            premodifier
                              adjective
                                PROMOTION PRICE
                                participle(trav, none, none, prpl)
                                  verb form(trav, none, none, prpl)
                                    LEX_VERBG(none, trav)
                                      VERBG(none, trav)
                                        "enduring"
                            rest premodifiers
                          noun kernel(sing)
                            LEX_NOUN(sing)
                              LNOUN(sing)
                                NOUN(sing)
```

```
                                          "work"
                      copula(sing, third)
                        to be(sing, third)
                          LEX_AUXV(sing, third)
                            AUXV(sing, third)
                              "must"
                          opt adverbs
                          LEX_TOBE(infi)
                            TOBE(infi)
                              "be"
                      participle(trav, none, from|on|into|in, papl)
                        verb form(trav, none, from|on|into|in, papl)
                          LEX_VERBP(from|on|into|in, trav)
                            VERBP(from|on|into|in, trav)
                              "built"
                      pref PP(from|on|into|in)
                      agent
                      opt circumstances
                opt circumstances
                  circumstance
                    PP
                      PP(like)
                        opt adverbs
                        LEX_PREPOS(like)
                          PREPOS(like)
                            "like"
                        noun phrase(sing, third, dat)
                          noun part(sing, third, dat)
                            article(sing)
                              LEX_ART(sing)
                                ART(sing)
                                  "a"
                            noun group(sing, third, dat)
                              noun kernel(sing)
                                LEX_NOUN(sing)
                                  LNOUN(sing)
                                    NOUN(sing)
                                      "machine"
                  opt circumstances
                    circumstance
                      adverb
                        LEX_ADVB
                          LADVB
                            ADVB
                              "full"
                    opt circumstances
                      circumstance
                        PP
                          PP(of)
                            opt adverbs
                            LEX_PREPOS(of)
                              PREPOS(of)
                                "of"
                            noun phrase(plur, third, dat)
                              noun part(plur, third, dat)
                                noun group(plur, third, dat)
                                  noun kernel(plur)
                                    LEX_NOUN(plur)
                                      LNOUN(plur)
                                        NOUN(plur)
                                          "shortcomings"
                      opt circumstances
  period
    "."
```

Connexor machinese for English (syntax tree output):



#	Text	Baseform	Syntactic relation	Syntax and morphology
1	Short-lived	short-lived	subj:>2	@SUBJ %NH A ABS
2	like	like	main:>0	@+FMAINV %VA V PRES
3	a	a	det:>4	@DN> %>N DET SG
4	machine	machine	obj:>2	@OBJ %NH N NOM SG
5	that	that	subj:>6	@SUBJ %NH <Rel> PRON
6	is	be	v-ch:>7	@+FAUXV %AUX V PRES SG3
7	used	use	mod:>4	@-FMAINV %VP EN
8	but	but	cc:>7	@CC %CC CC
9	not	not	cc:>7	@ADVL %EH NEG-PART
10	good	good		@PCOMPL-S %NH A ABS
11	enough	enough	ad:>10	@<AD-A %<E ADV
12	whilst	whilst	tmp:>13	@ADVL %EH ADV WH
13	promising	promise	tmp:>23	@-FMAINV %VA ING
14	to	to	pm:>15	@INFMARK> %AUX INFMARK>
15	be	be	obj:>13	@-FMAINV %VA V INF
16	better	good	comp:>15	@PCOMPL-S %NH A CMP
17	,	,		
18	an	an	det:>20	@DN> %>N DET SG
19	enduring	enduring	attr:>20	@A> %>N A ABS
20	work	work	subj:>21	@SUBJ %NH N NOM SG
21	must	must	v-ch:>22	@+FAUXV %AUX V AUXMOD
22	be	be	v-ch:>23	@-FAUXV %AUX V INF
23	built	build		@-FMAINV %VP EN
24	like	like	ha:>23	@ADVL %EH PREP
25	a	a	det:>26	@DN> %>N DET SG
26	machine	machine	pcomp:>24	@<P %NH N NOM SG
27	full	full	mod:>26	@<NOM %N< A ABS
28	of	of	mod:>27	@<NOM-OF %N< PREP
29	shortcomings	shortcoming	pcomp:>28	@<P %NH N NOM PL
30	.	.		
31	<p>	<p>		

10. *During the automatic customization (or training) of MSR-MT (see figure below), pairs of corresponding source and target sentences are parsed to produce graph-like structures called Logical Forms (LFs).*

Memory-based shallow parser demo:

Tagger output

During/IN the/DT automatic/JJ customization//NN (/(or/CC training/NN)/) of/IN MSR-MT//NNP (/(see/VB figure/NN below/IN)/) ,/, pairs/NNS of/IN corresponding/JJ source/NN and/CC target/NN sentences/NNS are/VBP parsed//VBN to/TO produce/VB graph-like//JJ structures/NNS called/VBD Logical//JJ Forms//NNS (/(LFs//NNP)/) ./.

Chunker output

{PNP [Prep During/IN Prep] [NP the/DT automatic/JJ customization//NN NP] PNP} (/(or/CC [NP training/NN NP])/) {PNP [Prep of/IN Prep] [NP MSR-MT//NNP NP] PNP} (/([VP see/VB VP] [NP figure/NN NP] [Prep below/IN Prep])/) ,/, [NP pairs/NNS NP] {PNP [Prep of/IN Prep] [NP corresponding/JJ source/NN and/CC target/NN sentences/NNS NP] PNP} [VP are/VBP parsed//VBN to/TO produce/VB VP] [NP graph-like//JJ structures/NNS NP] [VP called/VBD VP] [NP Logical//JJ Forms//NNS NP] (/([NP LFs//NNP NP])/) ./.

Subject/Object Detector output

{PNP [P During/IN P] [NP the/DT automatic/JJ customization//NN NP] PNP} (/(or/CC [NP training/NN NP])/) {PNP [P of/IN P] [NP MSR-MT//NNP NP] PNP} (/([VP_1 see/VB VP_1] [NP_1^{Object} figure/NN NP_1^{Object}] [P below/IN P])/) ,/, [$NP_2^{Subject}$ pairs/NNS $NP_2^{Subject}$] {PNP [P of/IN P] [NP corresponding/JJ source/NN and/CC target/NN sentences/NNS NP] PNP} [VP_2 are/VBP parsed//VBN to/TO produce/VB VP_2] [NP_2^{Object} graph-like//JJ structures/NNS NP_2^{Object}] [VP_3 called/VBD VP_3] [NP_3^{Object} Logical//JJ Forms//NNS NP_3^{Object}] (/([NP LFs//NNP NP])/) ./.

EP41R Parser:

Parse tree for the most probable analysis:

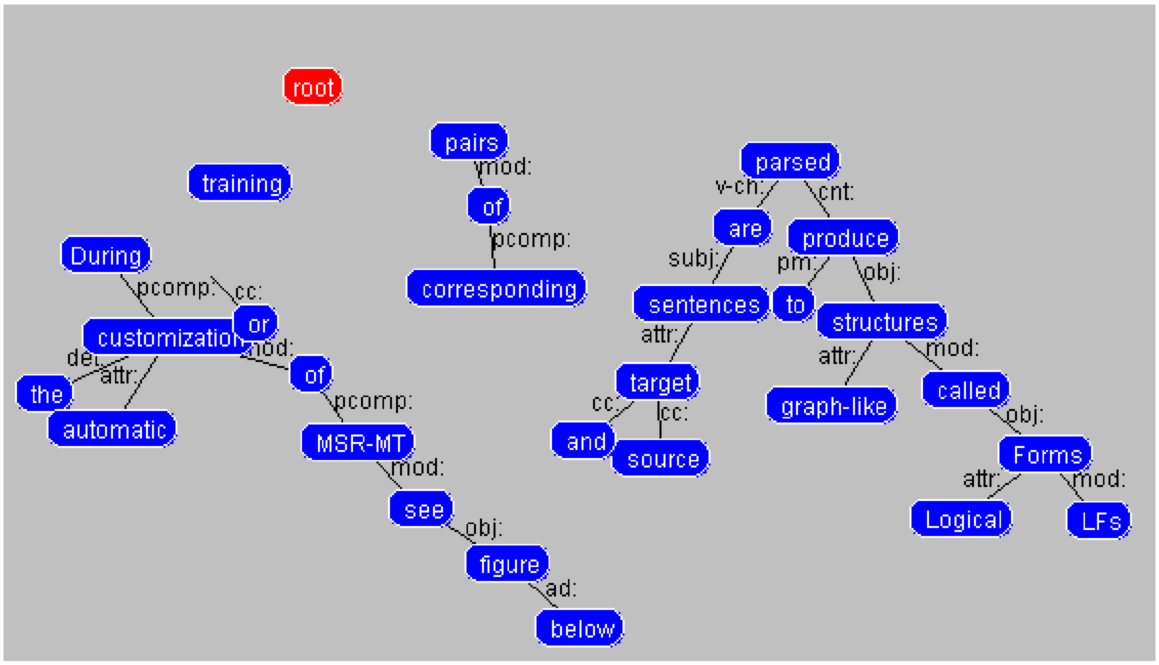
```
Warning: skipped unknown token "training)" at pos 39

Warning: skipped unknown token "below)" at pos 71

Warning: skipped unknown token "(LFs)" at pos 195



total number of parsings 0 (max 1)
total scan time 0.049
total parse time 0.000

segment
  NP
    noun phrase(sing, third, CASE)
      noun part(sing, third, CASE)
        article(sing)
          LEX_ART(sing)
            ART(sing)
              "the"
        noun group(sing, third, CASE)
          noun kernel(sing)
            premodifiers
              premodifier
                adjective
                  LEX_ADJE(abso)
                    LADJE(abso)
                      ADJE(abso)
                        "automatic"
              rest premodifiers
            noun kernel(sing)
              LEX_NOUN(sing)
                LNOUN(sing)
                  NOUN(sing)
                    "customization"


total number of parsings 1 (max 1)
total scan time 0.049
total parse time 0.000


total number of parsings 0 (max 1)
total scan time 0.049
total parse time 0.000


total number of parsings 0 (max 1)
total scan time 0.049
total parse time 0.001


total number of parsings 0 (max 1)
total scan time 0.049
total parse time 0.001


total number of parsings 0 (max 1)
total scan time 0.049
total parse time 0.001

segment
  sentence
    statement
      simple statement
        SVOC phrase
```

```
opt circumstances
subject(plur, third)
  NP(plur, third, nom)
    noun phrase(plur, third, nom)
      noun part(sing, third, nom)
        noun group(sing, third, nom)
          noun kernel(sing)
            LEX_NOUN(sing)
              LNOUN(sing)
                NOUN(sing)
                  "figure"
      coordinator
        LEX_CON(coo)
          CON(coo)
            ","
      noun phrase(plur, third, nom)
        noun part(plur, third, nom)
          noun group(plur, third, nom)
            noun kernel(plur)
              LEX_NOUN(plur)
                LNOUN(plur)
                  NOUN(plur)
                    "pairs"
            postmodifiers
              postmodifier
                PP(of)
                  opt adverbs
                  LEX_PREPOS(of)
                    PREPOS(of)
                      "of"
                  noun phrase(plur, third, dat)
                    noun part(plur, third, dat)
                      noun group(plur, third, dat)
                        noun kernel(plur)
                          premodifiers
                            premodifier
                              adjective
                                LEX_ADJE(abso)
                                  LADJE(abso)
                                    ADJE(abso)
                                      "corresponding"
                            rest premodifiers
                          noun kernel(plur)
                            premodifiers
                              premodifier
                                LEX_NOUN(sing)
                                  LNOUN(sing)
                                    NOUN(sing)
                                      "source"
                              rest premodifiers
                                coordinator
                                  LEX_CON(coo)
                                    CON(coo)
                                      "and"
                                premodifiers
                                  premodifier
                                    LEX_NOUN(sing)
                                      LNOUN(sing)
                                        NOUN(sing)
                                          "target"
                                  rest premodifiers
                            noun kernel(plur)
                              LEX_NOUN(plur)
                                LNOUN(plur)
                                  NOUN(plur)
                                    "sentences"
              rest postmodifiers
copula(plur, third)
  to be(plur, third)
    LEX_TOBE(plur, third)
      TOBE(plur, third)
```

```
              "are"
        participle(trav, none, none, papl)
          verb form(trav, none, none, papl)
            LEX_VERBP(none, trav)
              VERBP(none, trav)
                "parsed"
        pref PP(none)
        agent
        opt circumstances
          circumstance
            purpose
              "to"
              infinitive(trav, none, none)
                verb form(trav, none, none, infi)
                  LEX_VERBI(none, trav)
                    VERBI(none, trav)
                      "produce"
              OC phrase(trav, none, none)
                object
                  noun phrase(plur, third, acc)
                    noun part(plur, third, acc)
                      noun group(plur, third, acc)
                        noun kernel(plur)
                          premodifiers
                            premodifier
                              adjective
                                LEX_ADJE(ATTR)
                                  LADJE(ATTR)
                                    robust ADJE(ATTR)
                                      hyphenated form
                                        "graph-like"
                            rest premodifiers
                          noun kernel(plur)
                            LEX_NOUN(plur)
                              LNOUN(plur)
                                NOUN(plur)
                                  "structures"
                        participle(trav, none, none, papl)
                          verb form(trav, none, none, papl)
                            LEX_VERBP(none, trav)
                              VERBP(none, trav)
                                "called"
                        pref PP(none)
                        agent
                        opt circumstances
                  resultative part
                    NP(acc)
                      noun phrase(plur, third, acc)
                        noun part(plur, third, acc)
                          noun group(plur, third, acc)
                            noun kernel(plur)
                              premodifiers
                                premodifier
                                  adjective
                                    LEX_ADJE(abso)
                                      LADJE(abso)
                                        ADJE(abso)
                                          "logical"
                                rest premodifiers
                              noun kernel(plur)
                                LEX_NOUN(plur)
                                  LNOUN(plur)
                                    NOUN(plur)
                                      "forms"
                particle(none)
                C phrase(none)
                  opt circumstances
          opt circumstances
  period
    "."
```

```
total number of parsings 13 (max 1)
total scan time 0.049
total parse time 0.009
```

Connexor machinese for English (syntax tree output):



#	Text	Baseform	Syntactic relation	Syntax and morphology
1	During	during		@ADVL %EH PREP
2	the	the	det:>4	@DN> %>N DET
3	automatic	automatic	attr:>4	@A> %>N A ABS
4	customization	customization	pcomp:>1	@<P %NH N NOM SG
5	(	(		
6	or	or	cc:>5	@CC %CC CC
7	training	training		@APP %NH N NOM SG
				@A> %>N N NOM SG
8	)	)		
9	of	of	mod:>4	@<NOM-OF %N< PREP
10	MSR-MT	msr-mt	pcomp:>9	@<P %NH N NOM SG
11	(	(		
12	see	see	mod:>10	@+FMAINV %VA V IMP
13	figure	figure	obj:>12	@OBJ %NH N NOM SG
14	below	below	ad:>13	@ADVL %EH ADV
15	)	)		
16	,	,		
17	pairs	pair		@OBJ %NH N NOM PL
				@<P %NH N NOM PL
18	of	of	mod:>17	@<NOM-OF %N< PREP
19	corresponding	correspond	pcomp:>18	@<P-FMAINV %VA ING
20	source	source	cc:>22	@A> %>N N NOM SG
21	and	and	cc:>22	@CC %CC CC
22	target	target	attr:>23	@A> %>N N NOM SG
23	sentences	sentence	subj:>24	@SUBJ %NH N NOM PL
24	are	be	v-ch:>25	@+FAUXV %AUX V PRES
25	parsed	parse		@-FMAINV %VP EN
26	to	to	pm:>27	@INFMARK> %AUX INFMARK>
27	produce	produce	cnt:>25	@-FMAINV %VA V INF
28	graph-like	graph-like	attr:>29	@A> %>N A ABS
29	structures	structure	obj:>27	@OBJ %NH N NOM PL
30	called	call	mod:>29	@-FMAINV %VP EN
31	Logical	logical	attr:>32	@A> %>N A ABS
32	Forms	form	obj:>30	@OBJ %NH N NOM PL
33	(	(		
34	LFs	lfs	mod:>32	@NH %NH <?> N NOM SG
35	)	)		
36	.	.		
37	<p>	<p>		

ANALYSIS

The ten test sentences reveal some of the challenges of the three tested parsers. Some features tested include:

- Ambiguity, local and global
- Well-formed nonsense
- Ill-formed sentence
- Misspellings
- Language acts
- Machine-generated text
- Non-sequitur/ tr verb used as int v.
- Deeply embedded clauses
- Technical jargon
- Parentheticals

The Connexor parser did an excellent job on the global ambiguity example (sentence 1) – It determined that the duck in “made her duck” was the infinitive verb form, not the noun duck. It did guess that “Flying” in Flying planes” is a present participle modifying the noun “planes.” (in Connexor-speak, that’s a “premodifier of a nominal.” The EP41R fared poorly, deciding early on that “planes” was a verb. The MBT tagger performed somewhere in the middle: “duck” is a noun and flying somehow is a noun yet it did capture “made” as a verb. As for local ambiguity (example 2), all three parsers perform adequately. The parsers perform differently with respect to verb tense: Connexor assumes the verb is past tense while MBT assumes it is present. The EP41R does not appear to indicate any tense information. While I think in text it is more likely that the verb is past instead of present tense, there is no easy way to choose one or the other. The fact that Connexor picks the more likely choice may be of interest.

I expected that all three parsers would perform well with example 3, though the EP41R places “drink” as a noun. It begins to become apparent that the EP41R “jumps to conclusions:” it decides what part of speech a word is before inspecting the words following it, and somehow decides that “ghosts” is a verb instead of a noun, the more likely possibility. Inspection of later words would have cleared this up by showing other possibilities for verbs. The other two parsers parse the example correctly.

Since example 4 is an ill-formed statement, the point of interest here surrounds handling of “bad grammar”: does the parser try to force a working structure, and if it does not, does it somehow elegantly represent the point of difficulty? Both the MBT and the EP41R assume the sentence is well-formed and treat it accordingly, in a way that seems “correct.” Yet the Connexor gets it right: not only does it capture the ill-formed-ness, but it represents the ill-formed structure in a reasonable way, namely by separating the phrase “to home” from the well-formed “Sally is going probably.” Other “erroneous” sentences, examples 5 and 6, were perhaps a bit too unfair. I should not have expected any of the parsers to know how to parse misspellings or inarticulate speech, and none of them did know. I did hope that at least one would try to guess at a version with the misspellings repaired.

Example 7 is an excellent example of a non-sequitur expressive speech act that is more frequently though not exclusively found in speech rather than text. What surprised me was something I was not looking for: the EP41R had no idea what the negation contraction was. The Connexor and MBT perform well at handling the example, though I question whether it should not show the fact that the transitive verb “say” is fine in an intransitive form.

Example 8 was a test borne from plain curiosity about machinic-language. The MBT and Connexor both perform quite well: each performs consistently from the example to on the more obvious but similarly structured “I am woman.” I find little surprise that the EP41R parsed the example correctly given that it was so simple.

Examples 9 and 10 tested the parsers’ abilities to handle long sentences with deeply embedded phrases, parentheticals, antiquated language, and jargon. These two examples undoubtedly posed a large number of difficult problems: all of the parsers were expected to fail on all examples. I found it interesting how the Connexor failed to handle the phrase “but not good enough.” I tested a number of examples to locate the problem: “I am good enough,” “I am not good enough,” “I am new and good enough, “I am new but good enough” all parse perfectly well, yet “I am new but not good enough” fails just as it does in the present example: “good enough” is split off, thus revealing that the system fails to handle negated embedded phrases. These two examples were on the whole much too difficult even for the human to understand reliably well, and the embedding was too deep, thus making it too difficult for the parsers to recognize even where the divide between the main NP and VP should be. . I was surprised that all three recognized the word “whilst.”

It seemed clear from the very start that the Connexor would outperform the other parsers, and that EP41R would follow behind the other two. Sentence 1 was in my estimation the best test of current parsers: it was a moderately difficult test that required some understanding of embedded phrases and handling both multiple verbs-looking words and multiple meanings. Ambiguity seems to continue to be a focus area of parsing research and for good reason: it is one of the features of language that seems to run contrary to logic-based divide-and-conquer paradigms. At the same time it is a semantic feature that can be resolved by co-occurrence frequencies with unambiguous terms in the same sentence: the sentence can be its own context. Ambiguity might be thought of as a rudimentary form of a language problem that requires resolution on some level: misspellings, embedded phrases, ill-formed-ness, nonsense, technical jargon, parentheticals, can all be treated as decision nodes connected to multiple possibilities. It seems no surprise that the performance of the three parsers ranked over the ten sentences the same as they did with the first example.

I would like to see parsers offer multiple versions of sentence parsings—I’d like them to show their decisions and rejects. To wit, if a sentence is ambiguous when taken out of context, and the ambiguity indicates two possible readings of that sentence< I’d like to see that parser show both instead of merely picking one. Such possibilities could be represented in a tree with nodes representing decision points. I would also like to see, in the spirit of representing multiple alternatives, a parser handle misspellings and represent possibilities with possible correct spellings. None of the parsers even attempted to handle misspelled words. Finally I hope to see a NLP parser that may be able to better handle text representations of dialogue and other more ‘realistic” examples of language.