Time Flies (Solution)

Overview

This puzzle is in the form of a phrase structure grammar having a set of phrase structure rules and a lexicon specifying the syntactic categories of 64 words.

Some of the words may be familiar from famous example sentences in linguistics. Indeed, all of the words are in a set of 10 more-or-less classic sentences, given here without capitalization or punctuation to facilitate parsing and because capitalization and punctuation disambiguate some of these sentences.

- an american flag was hanging in front of every building[1]
- buffalo buffalo buffalo buffalo buffalo buffalo buffalo buffalo[2]
- colorless green ideas sleep furiously[3]
- daddy what did you bring that book that i don't want to be read to out of up for[4]
- every farmer who owns a donkey beats it[5]
- flying planes can be dangerous[6]

[1]Hirschbühler, Paul. 1982. VP-deletion and across-the-board quantifier scope. In *Proceedings of NELS 12*, 132–139. GLSA, University of Massachusetts, Amherst. Other variants exist but do not appear to be as widespread. Although not as widely known outside the academic literature, this is a classic example illustrating a scope ambiguity, which is a semantic ambiguity and also possibly a syntactic ambiguity, depending on your theory of the syntax-semantics interface. The sentence has two possible readings, one in which a single flag is hanging in front of all of the buildings, and one in which each building has its own flag hanging in front of it. Neither reading is relevant to this puzzle because the given grammar does not care about real English.

[2]See `http://www.cse.buffalo.edu/~rapaport/buffalobuffalo.html` for a history of this sentence. This a classic example which is both ambiguous and hard to parse due to the use of three senses of *buffalo*, though it can be disambiguated by capitalization and punctuation. Variations exist with other numbers of *buffalo*s, but the version with eight *buffalo*s appears to be the most common. Regardless, for any $n \geq 1$, the sentence consisting of n instances of buffalo has an unambiguous parse under the given grammar.

[3]Chomsky, Noam. 1956. Three models for the description of language. *IRE Transactions on Information Theory* (2): 113–124. Though not ambiguous, this is a classic sentence showing that a sentence can be syntactically well-formed (and thus parsable) while being semantically incoherent.

[4]Taken from Pinker, Steven. 1994. *The language instinct: How the mind creates language.* New York: William Morrow. This classic sentence illustrates the possibility of coherently ending a sentence with five prepositions, defying the unfounded but oft-repeated prescriptivist prohibition against ending a sentence with a preposition. For our purposes, the given grammar doesn't care that the five words are prepositions in real English and classifies them in its own way. Pinker describes this as "an old grammarian's saw about how a sentence can end in five prepositions". There is an earlier citation from E. B. White in 1962 that gives the sentence without the initial *daddy*. Ideally, this would be the version to use, being older and slightly more widely cited, but unfortunately it does not fit the ordering scheme, so we have opted for the Pinker version, which is still widely cited. The grammar is written so that solvers who think the sentence is the other version will realize that they are missing a word at the beginning, which should point them to the Pinker version as the only possible choice.

[5]Geach, Peter Thomas. 1962. *Reference and generality.* Ithaca: Cornell University Press. Other variations exist in the literature that seek to avoid the animal cruelty, but this is the original sentence. Although classic, this sentence is horrible. We do not advise anyone, farmers or otherwise, to beat donkeys, except perhaps in the sense of beating a donkey in a race. The sentence illustrates a complication for the syntax-semantics interface, specifically anaphor resolution.

[6]Chomsky, Noam. 1955. *The logical structure of linguistic theory.* Published in 1975. New York: Plenum Press. This classic sentence illustrates a syntactic ambiguity that cannot be distinguished in terms of constituency

- the gostak distims the doshes[7]
- the horse raced past the barn fell[8]
- i saw the man with the binoculars[9]
- james while john had had had had had had had had had had had a better effect on the teacher[10]

The title refers to another famous sentence: *Time flies like an arrow; fruit flies like a banana.* Indeed, there are a lot of arrows in the rules; this is the standard notation, but it fits thematically.

Many of the sentences are examples of syntactic ambiguity with multiple parses or are difficult to parse due to repetition of words or garden-path effects. Other sentences illustrate the possibility of syntactic well-formedness without semantic coherence, or difficulties in mapping between syntax and semantics.

Since parsing is a theme in most of the sentences, this suggests parsing the sentences using the given grammar. Although the grammar is junk and doesn't match the actual structure of English at all, it does generate all of these sentences, and in many cases, it should be clear that the grammar generates multiple possible parses for sentences. Each sentence uses exactly one of the top-level rules uniquely, and as a confirmation, the first letter of the first non-article word in the sentence matches the phrasal category (for example, flying planes can be dangerous matches the rule `S` $\rightarrow$ `FP VP` rule); the sentences are also ordered from A through J, as a check that you have the right version of each sentence.

To solve the puzzle, count the number of parses the grammar produces for each sentence (details are provided in the next section) and map this to a letter (1 parse $\rightarrow$ **A**, 2 parses $\rightarrow$ **B**, etc.). These can be ordered by sorting the sentences in alphabetical order. Doing this yields the clue phrase **GALA OR FUJI**. These are two common varieties of apples, so the final answer is **APPLE**. The fruit flies in the sentence referenced by the title may prefer a banana, but if one is not available, an apple would probably do just as well. Also, apples seem to be popular fruits to shoot arrows at. Sometimes this involves placing an apple on someone's head. We do not advise engaging in this activity.

alone, as *flying planes* forms a constituent in either reading. Under the given grammar, however, this need not be a constituent in all parses.

[7]Ingraham, Andrew. 1903. *Swain School lectures.* Chicago: Open Court. This classic sentence illustrates the possibility of forming syntactically valid sentences from novel words not in the lexicon. In our grammar, the nonsense words are actually in the lexicon, though not defined.

[8]Bever, Thomas G. 1970. The cognitive basis for linguistic structures. In R. Hayes (ed.), *Cognition and language development*, 279–362. New York: Wiley. This classic sentence looks ungrammatical but is actually parsable (kind of) by treating *raced past the barn* as a passive relative clause modifier of *horse*. It illustrates the garden path effect, where a more likely parsing is picked early on but crashes later in the sentence. Kind of like the poor horse following the garden path beyond the barn. (What is with these example sentences and equines getting hurt?!) In our grammar, the sentence is actually grammatical without the final fell, and the ambiguity is in where to attach it, which nicely fits the parsing difficulty of the original sentence.

[9]The origin of this sentence, and its many variants, is unclear. This version appears to be the most common and is the only one that can be generated with the given lexicon. This is a classic sentence illustrating an attachment ambiguity: *with the binoculars* could be a verbal modifier (looking through the binoculars) or a nominal modifier (seeing someone who is holding binoculars). Under our given grammar, there are even more crazy ways to parse this sentence!

[10]The origin of this sentence, and similar ones with different names, is unclear. This version seems to be the most common by far, and is the only one that can be generated with the given lexicon, unless you switch james and john, in which case the resulting sentence still has the same parses and there is no effect on the alphabetical ordering of the sentences. This is a classic sentence (or pair of sentences) obfuscated by its lack of punctuation. Since our grammar doesn't care about punctuation, this is not a problem.

Parsing the sentences

Counting parses can be done by writing a program to do so, using a dynamic programming algorithm that counts the number of parses a subsequence has for any given root category. Alternatively, one can treat it as a combinatorial logic puzzle and figure it out by hand, which should be a straightforward task made clearer by the category labeling scheme. The rest of this document is a detailed explanation of how to do so.

Since each word has a unique category label, we can simplify the process by first replacing the words in each sentence with terminal categories and counting the number of parses for a sequence of categories. This also brings out similar substructure across the sentences more clearly. Also, it is useful to note that every phrase consists of at least one word, since there is no rule that generates a null terminal.

This process reveals that the sentences consist of parts where the terminals are all `A`, `B`, or `C` and sequences consisting solely of `X`, `Y`, or `Z`, with several long sequences of `X` in several places. There is a reason for this, which becomes clear when we try parsing the sentences using the rules.

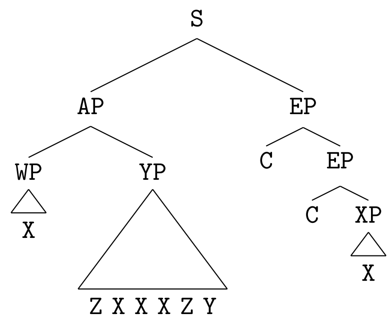
Looking at the rules reveals that `X`, `Y`, and `Z` can only be generated within a subtree labeled `UP`, `VP`, `WP`, `XP`, `YP`, or `ZP`, and that within these subtrees, there is no way to generate the terminals `A`, `B`, or `C`. So all of the sequences of `X`, `Y`, and `Z` are completely isolated within these subtrees.

Next, trying to parse the example sentences using the rules reveals that the overall structure of each sentence is unambiguous, with the exception of internal configurations of subtrees labeled `UP`, `VP`, `WP`, `XP`, `YP`, or `ZP`. The next section gives the overall parse trees for the sentences, without expanding the special subtrees just mentioned. We see that many instances of these subtrees are identical across the sentences, so figuring out each instance will give us a parse count for other occurrences of that subtree.

General parse trees

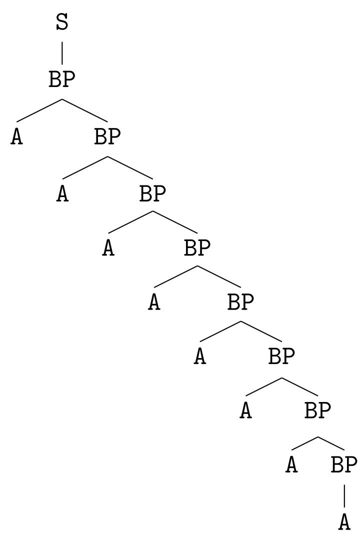
- an american flag was hanging in front of every building

  Terminals: `X Z X X X Z Y C C X`



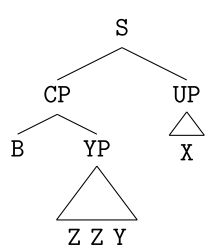
- buffalo buffalo buffalo buffalo buffalo buffalo buffalo buffalo

  Terminals: `A A A A A A A A`



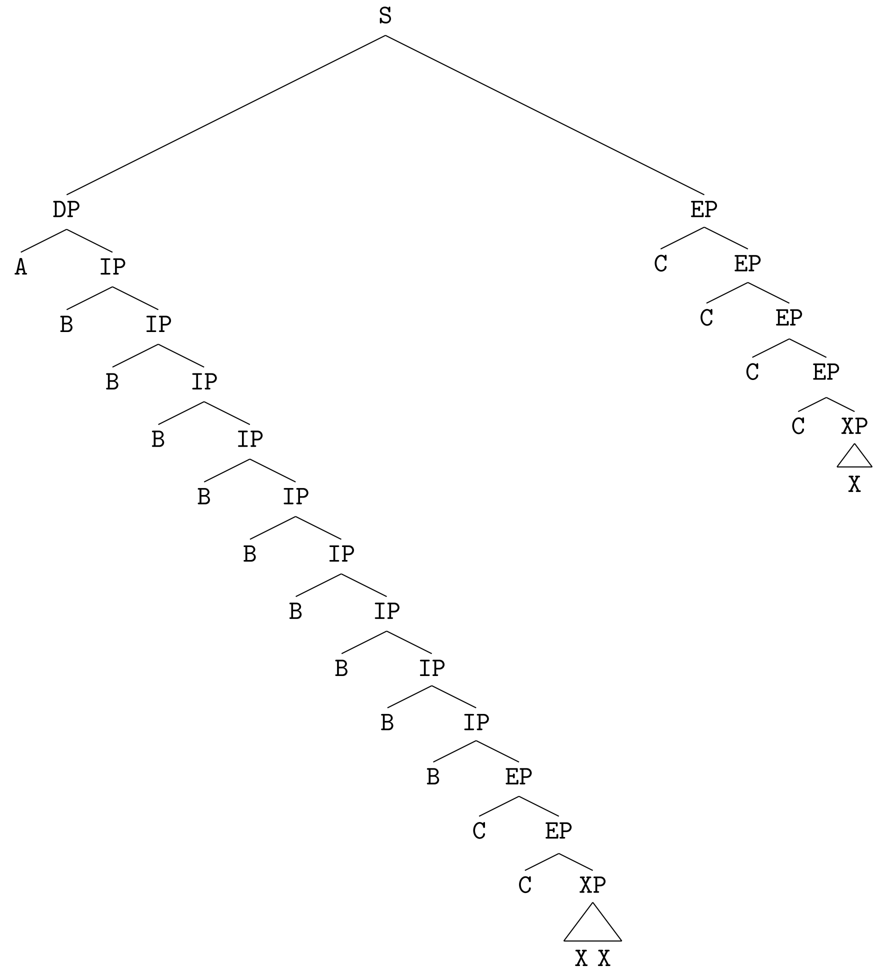
- colorless green ideas sleep furiously

  Terminals: `B Z Z Y X`



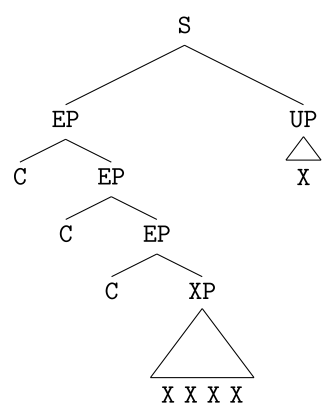
- daddy what did you bring that book that i don't want to be read to out of up for

  Terminals: `A B B B B B B B B B C C X X C C C C X`



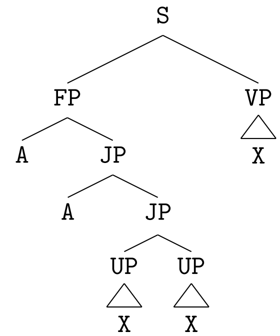
- every farmer who owns a donkey beats it

  Terminals: `C C C X X X X X`



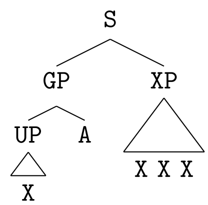
- flying planes can be dangerous

  Terminals: `A A X X X`



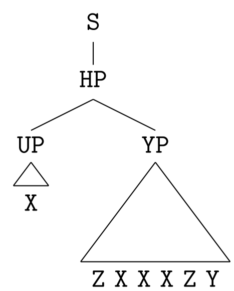
- the gostak distims the doshes

  Terminals: `X A X X X`



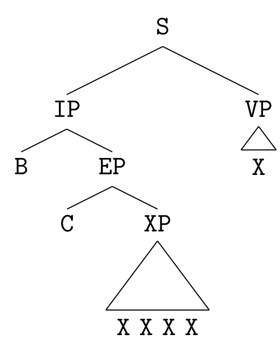
- the horse raced past the barn fell

  Terminals: `X Z X X X Z Y`



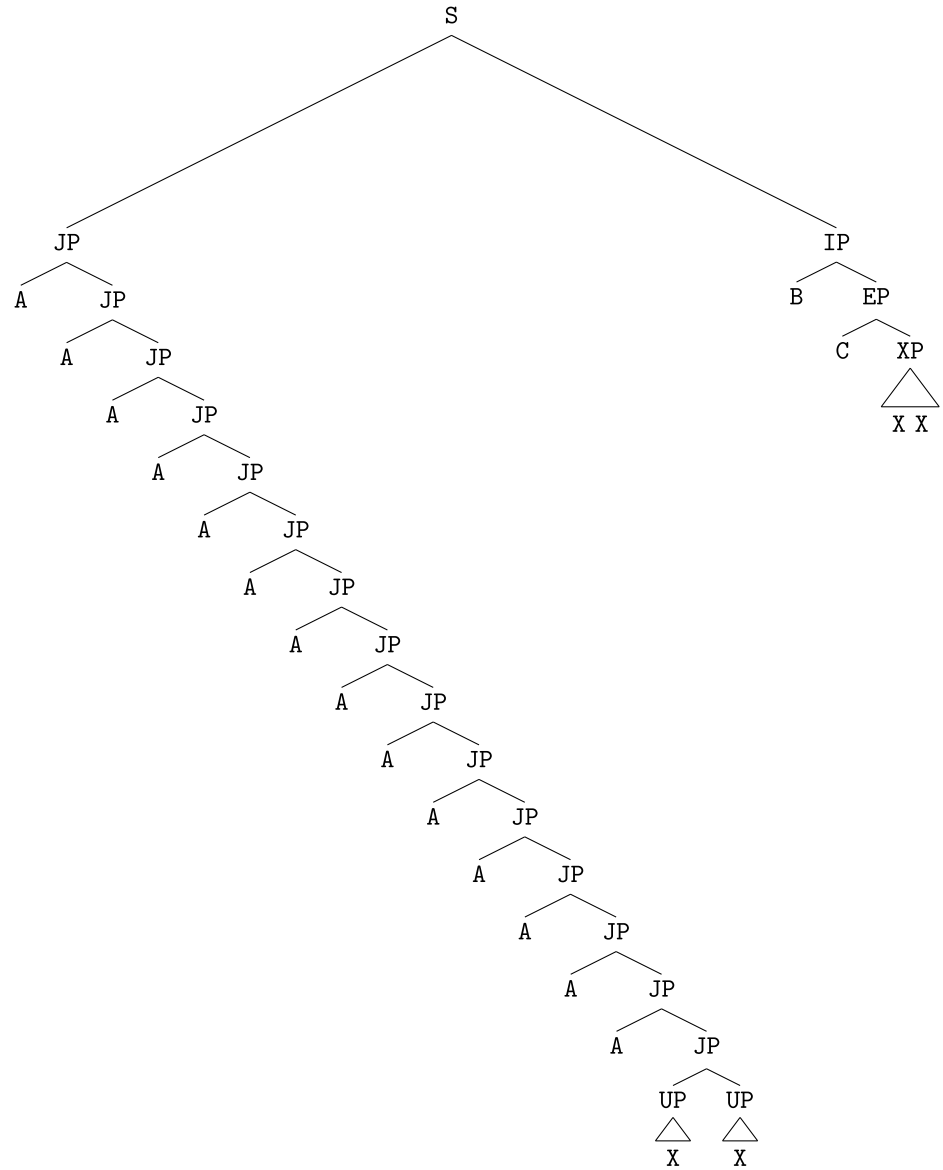
- i saw the man with the binoculars

  Terminals: `B C X X X X X`



- james while john had had had had had had had had had had had a better effect on the teacher

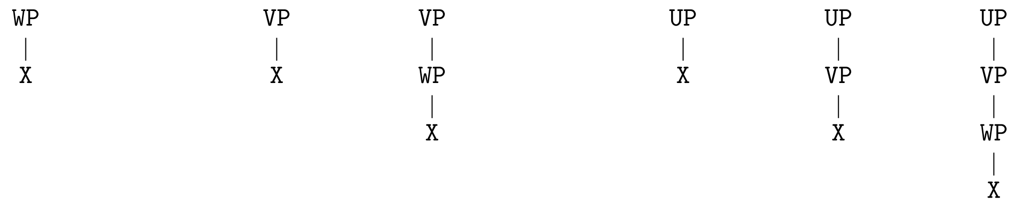
  Terminals: A A A A A A A A A A A A A A X X B C X X



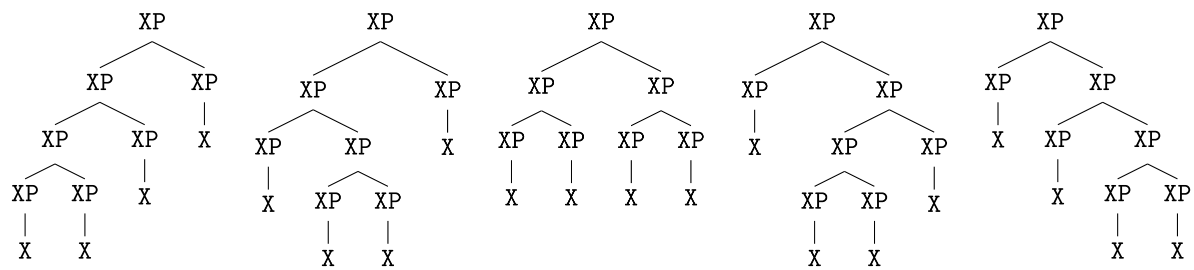
Three classes of parsing ambiguities

We can now count the parses for each possible subtree. These fall into three classes.

The first is an ambiguity caused by a particular chain of unary-branching categories `UP`, `VP`, and `WP`. Each one can be rewritten by the one after it, as well as rewriting to `X` directly. This means that `WP` has 1 parse, `VP` has 2 parses, and `UP` has 3. Here are all of the possible parses.

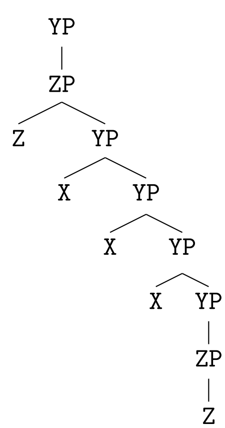


The second is a binary-branching grouping ambiguity. This happens whenever an `XP` generates a sequence of n instances of `X`. The number of parses is the $(n-1)$st Catalan number: 1 `X` means 1 parse, 2 `X`'s mean 1 parse, 3 `X`'s mean 2 parses, and 4 `X`'s mean 5 parses. (There is no need for higher numbers in this puzzle.) Here are all five parses for a sequence of four `X`'s:



The third is an adjunct attachment ambiguity. This is because a `Y` can attach to multiple possible `YP`'s or `ZP`'s preceding it. A `YP` subtree generates a sequence of `X`'s and `Z`'s, followed by a limited number of `Y`'s. In this puzzle, there are only three instances of `YP` subtrees, and they have the form of `Z` $\texttt{X}^n$ `Z Y`, for $n \geq 0$.

If we disregard the `Y`, the remaining sequence can be parsed unambiguously as a `YP`. Here is an example for the subtrees found in two of the sentences, where there are 3 `X`'s.



Now any of the `YP` or `ZP` nodes can be expanded to attach a `Y`, in the following ways:

Since there is only one `Y`, we just need to count the number of `YP`'s and `ZP`'s in the chain to the left of it to determine the number of parses. For this example, there are 7 parses. In general, with n `X`'s in the given pattern, there are $n+4$ parses. This puzzle only has instances of this pattern for $n = 0$ and $n = 3$.[11]

The special subtrees are distinct, so configurations in any subtree have no effect on the possible parses within another subtree. This means that the total number of parses for a tree is simply the product of the number of parses for each of the special subtrees.

This table gives the products for the sentences:

Sentence	Subtree parses	Total parses	Letter
an american flag ...	1, 7, 1	7	**G**
buffalo buffalo buffalo ...	(none)	1	**A**
colorless green ideas ...	4, 3	12	**L**
daddy what did ...	1, 1	1	**A**
every farmer who ...	5, 3	15	**O**
flying planes can ...	3, 3, 2	18	**R**
the gostak distims ...	3, 2	6	**F**
the horse raced ...	3, 7	21	**U**
i saw the ...	5, 2	10	**J**
james while john ...	3, 3, 1	9	**I**

[11] In general, `YP` can generate sequences outside of this pattern, but accounting for these is not necessary for this puzzle. If you are interested, `YP` can generate sequences consisting of any number of `X`'s and `Z`'s in any order, followed by a `Z`, followed by any number of `Y`'s. For n `X`'s, m `Z`'s, and k `Y`'s, the number of possible attachment sites is $n + 2m$, and the number of distinct ways to assign k `Y`'s to these attachment sites is the multiset coefficient

$$\left(\!\!\binom{n+2m}{k}\!\!\right) = \binom{n+2m+k-1}{k} = \frac{(n+2m+k-1)!}{k!(n+2m-1)!}.$$