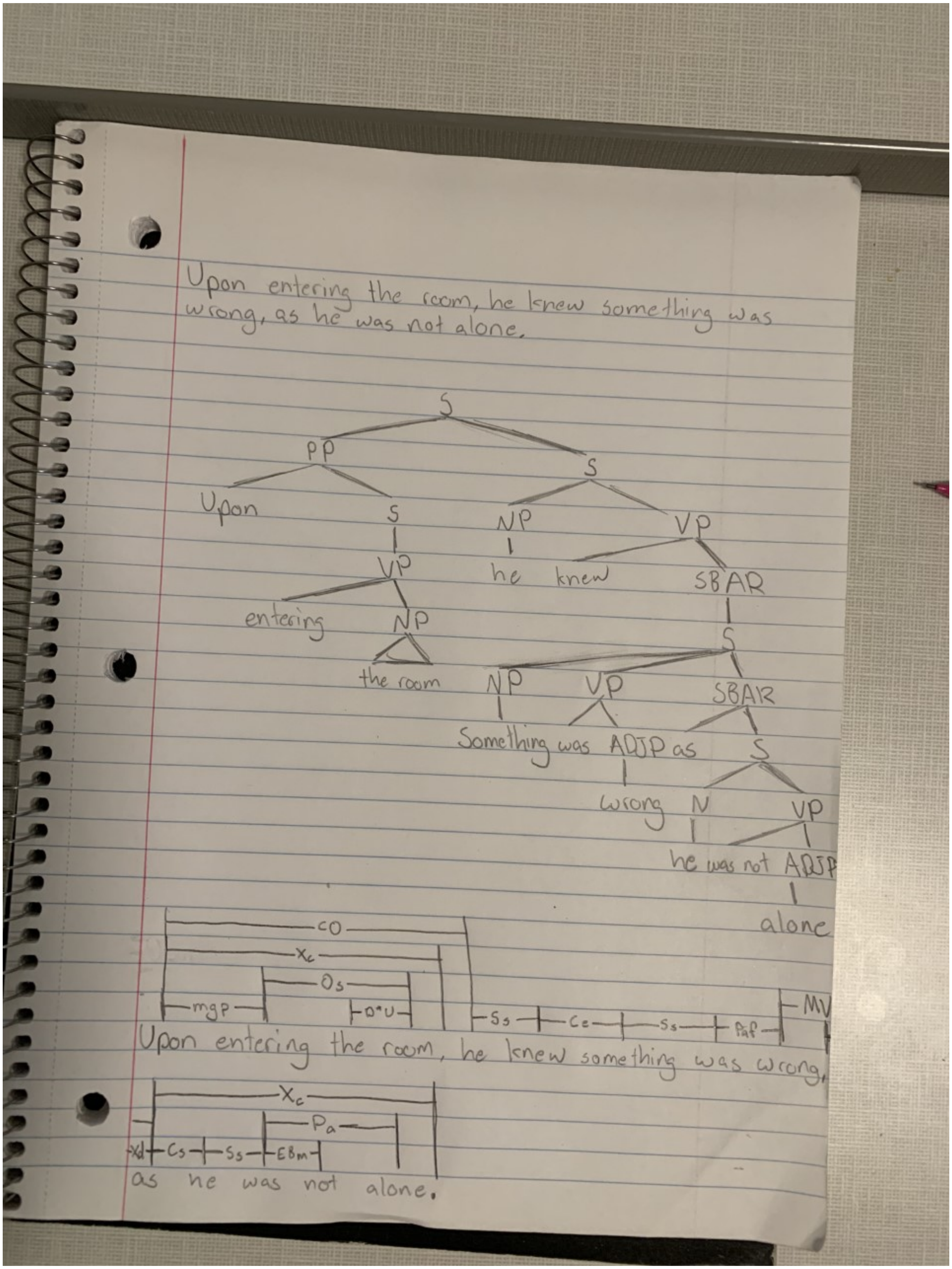
PSG Tree / Dependency Parse



Upon entering the room, he knew something was wrong, as he was not alone.

Phrase Terms

- **S** - simple declarative clause
- **PP** - Prepositional Phrase
- **VP** - Verb Phrase
- **NP** - Noun Phrase
- **SBAR** - Clause introduced by a possibly empty subordinating conjunction
- **ADJP** - Adjective Phrase

Dependency Relations

- **CO** - Openers
- **Xc** - word connected to comma on right
- **Os** - transitive verb to singular noun
- **Mgp** - connecting noun to past participle conjunction
- **D*u** - Connects determiner to noun
- **Ss** - Singular noun to singular verb
- **Ce** - verbs that take clausal complements
- **Paf** - connects certain verbs to predicative adjectives
- **MVp** - connects verbs and adjectives to modifying phrases like adverbs
- **EBm** - connects adverbs to forms of "be" before an object, adjective, or prepositional phrase

SRL Parse

- Entering
  - ARG0 - he
  - ARG1 - the room
- Knew
  - ARG0 - he
  - ARG1 - something was wrong
  - ARGM-TMP - Upon entering the room
  - ARGM-CAU - as he was not alone
- Was
  - ARG1 - something
  - ARG2 - wrong
- Was
  - ARG1 - he
  - ARG2 - alone
  - ARGM-NEG - not
- **Modifiers**
  - TMP - temporal, when verb took place
  - CAU - Causal, why verb took place
  - NEG - Negative, inverts next argument

Arguments

The first argument is generally what is committing the verb, and the second argument is how the verb is committed. The rest of the arguments are all grammatically optional context and details, such as when or why the verb was committed.

Pros/Cons of each Parse Type

The Pros and Cons of the first Parse Type, the PSG tree, is as follows. The Pros are that they clearly label what grammatical term each word is, and what grammatical term each word is a part of, thus making a distinct hierarchy of terms. The downside is that while it is clear which terms are grouped together, it isn't clear how the grouped terms are connected. The next Parse type, the dependency parse, is the opposite. The upside of it is that it highlights whatever connections/dependencies exist between any given pair of terms. However, the downside is that it doesn't give any kind of structure beyond a given pair of terms, and it doesn;t directly give info on what each word is, though that can be deducted based on the connections. The third Parse type, SRL Parsing, has its own Pros and Cons. The Pros are that it defines a specific type of connection it will be recording, that being the verbs and the arguments that provide them context, and clearly labels each term in relation to the verb. The downside is that in terms of efficiency, each verb reuses arguments from other nodes, making it somewhat redundant, and that only verbs are independently defined. All other arguments are not defined by what they are, but what they are relative to the verb. For example, whereas in some other Parse a token might be a Noun that is part of a Noun Phrase, in this it would simply be argument[0], losing a lot of its own meaning.