CSC344 BNF Assignment

Learning Abstract:

For this task, I assembled BNF grammars in the shapes of mathematical expressions, color outcomes, and other things, and then I came up with a straightforward definition of what BNF meant in English. Problem sets that required generating the parse tree-correlating grammatical descriptions were a difficulty for me.

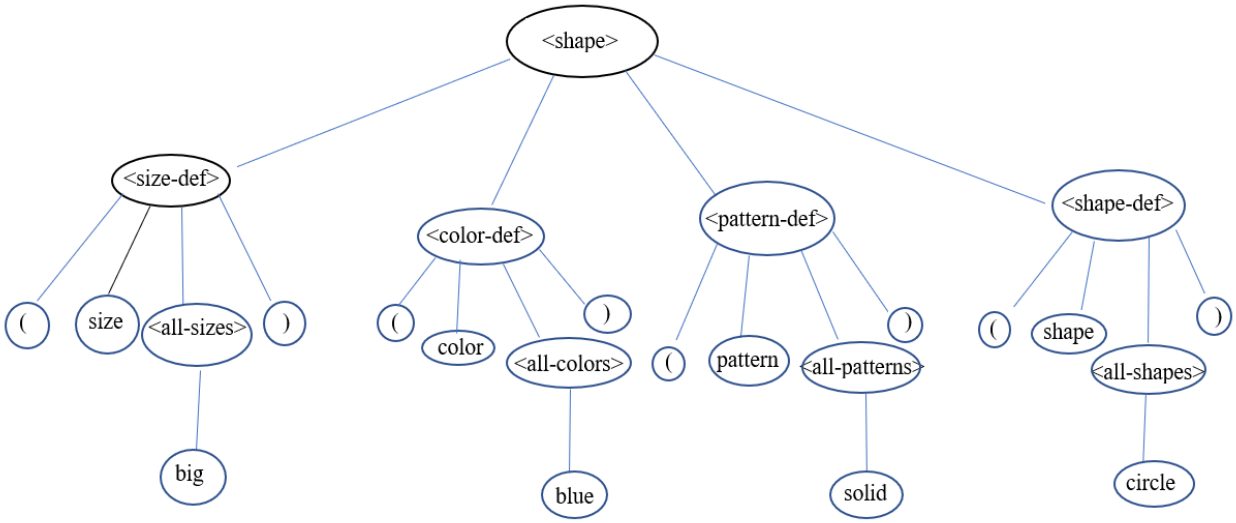
Problem 1 - Shapes

1) **Grammar description of the Shapes language**

<shapes> : : = <size-def> <color-def> <pattern-def> <shape-def>
<size-def> : : = (size <all-sizes>)
<color-def> : : = (color <all-colors>)
<pattern-def> : : = (pattern <all-patterns>)
<shape-def> : : = (shape <all-shapes>)
<all-sizes> : : = small | medium | big
<all-colors> : : = red | blue | yellow
<all-patterns> : : = stripped | dotted | solid
<all-shapes> : : = circle | square | triangle

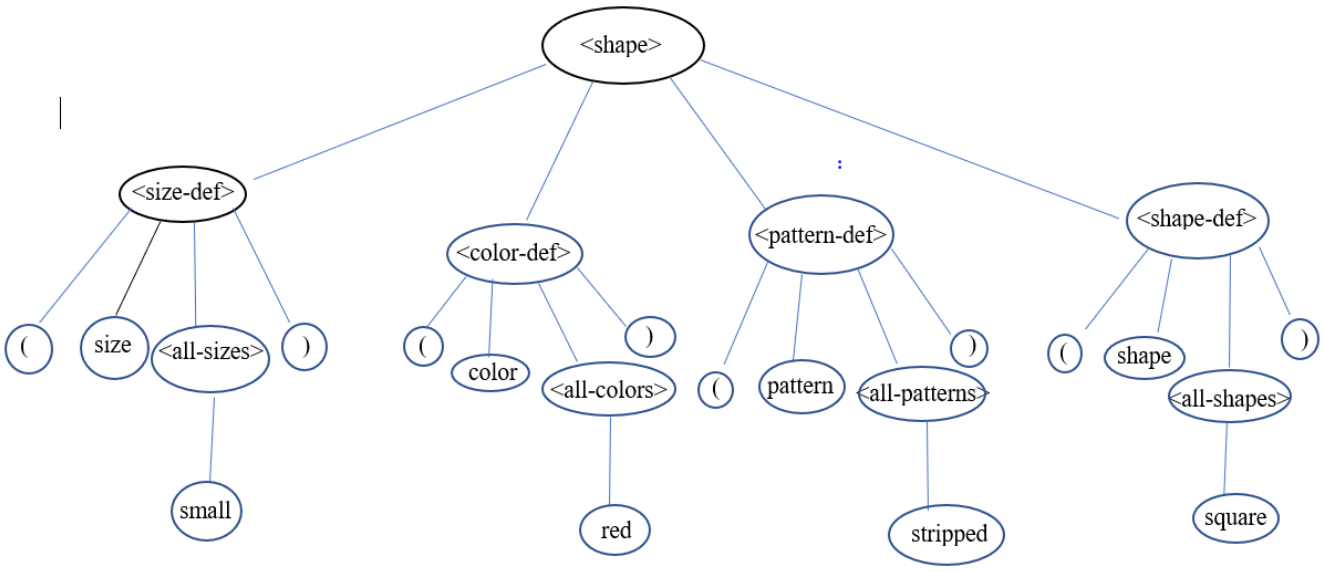
2) **Create parse tree:**

((size big) (color blue) (pattern solid) (shape circle))



3) **Create parse tree:**

((size small) (color red) (pattern striped) (shape square))



Problem 2 - SQN (Special Quaternary Numbers)

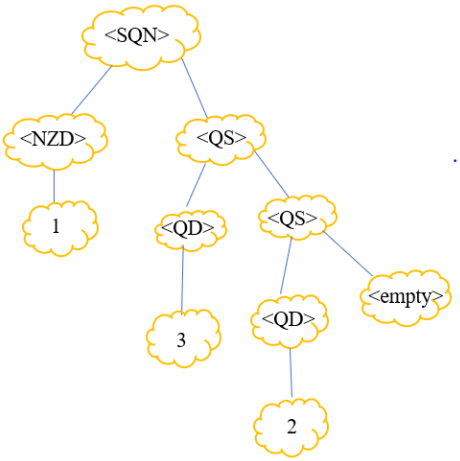
1. **Grammar Description of SQN**

<SQN> : : = 0 | <NZD> <QS>
<NZD> : : = 1 | 2 | 3
<QS> : : = <empty> | <QD>
<QD> : : = 0 | 1 | 2 | 3

NZD - non zero digit
QS - quaternary sequence
QD - quaternary digit

2. Parse Tree for: 0

3. Parse Tree for: 132



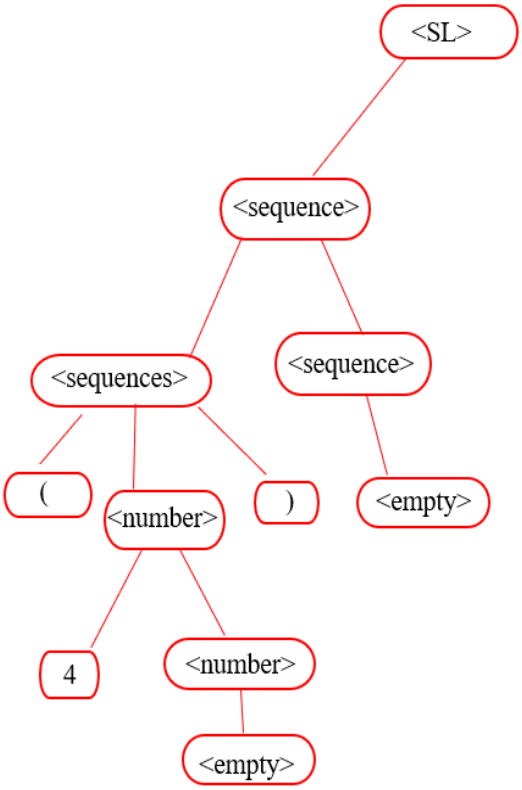
3. You cannot draw a parse tree, consistent with the BNF grammar that you crafted, for the string: 1223 because we can't have two adjacent occurrences of the same quaternary digit.

Problem 3 - Fours

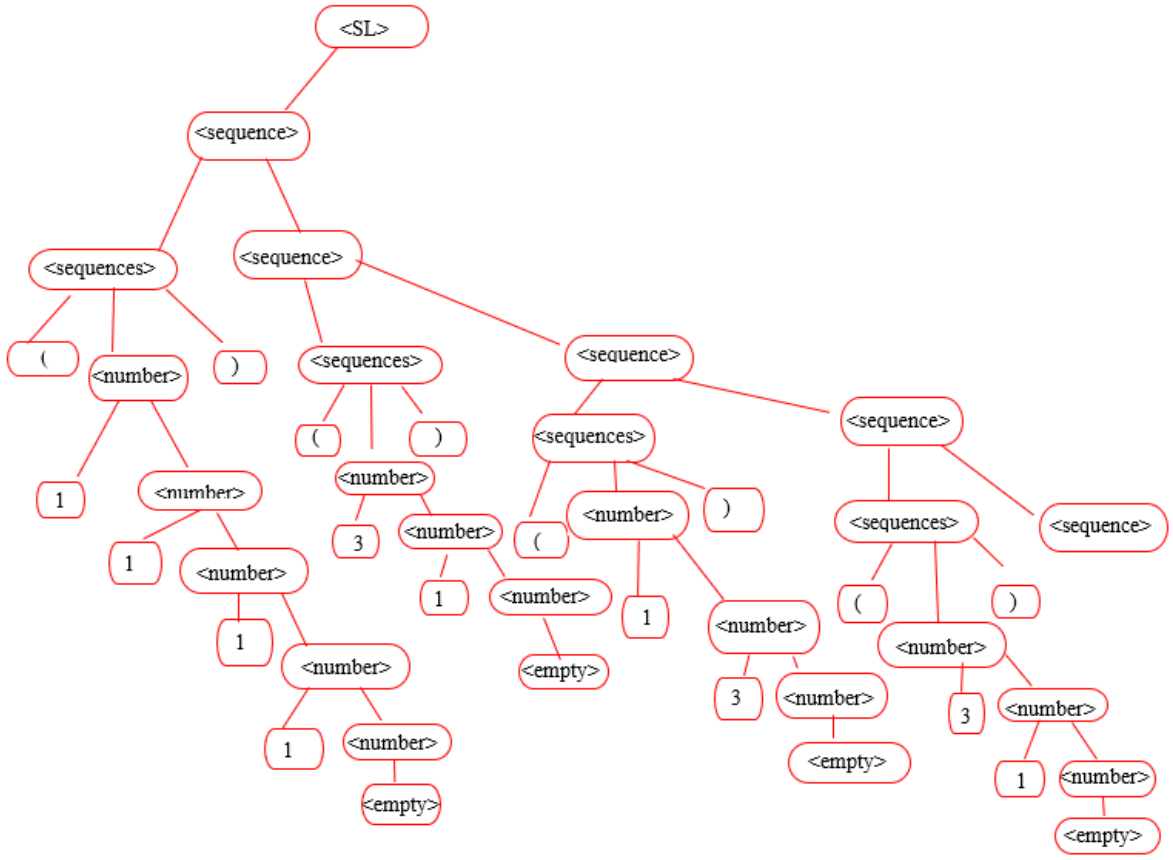
1. **Grammar Description of Fours**

<SL> : : = <empty> | <sequences>
<sequence> : : = <empty> | <sequence> < sequences>
<sequences> : : = (<number>)
<number> : : = <empty> | 1 <number>| 2 <number> | 3 <number> | 4 <number>

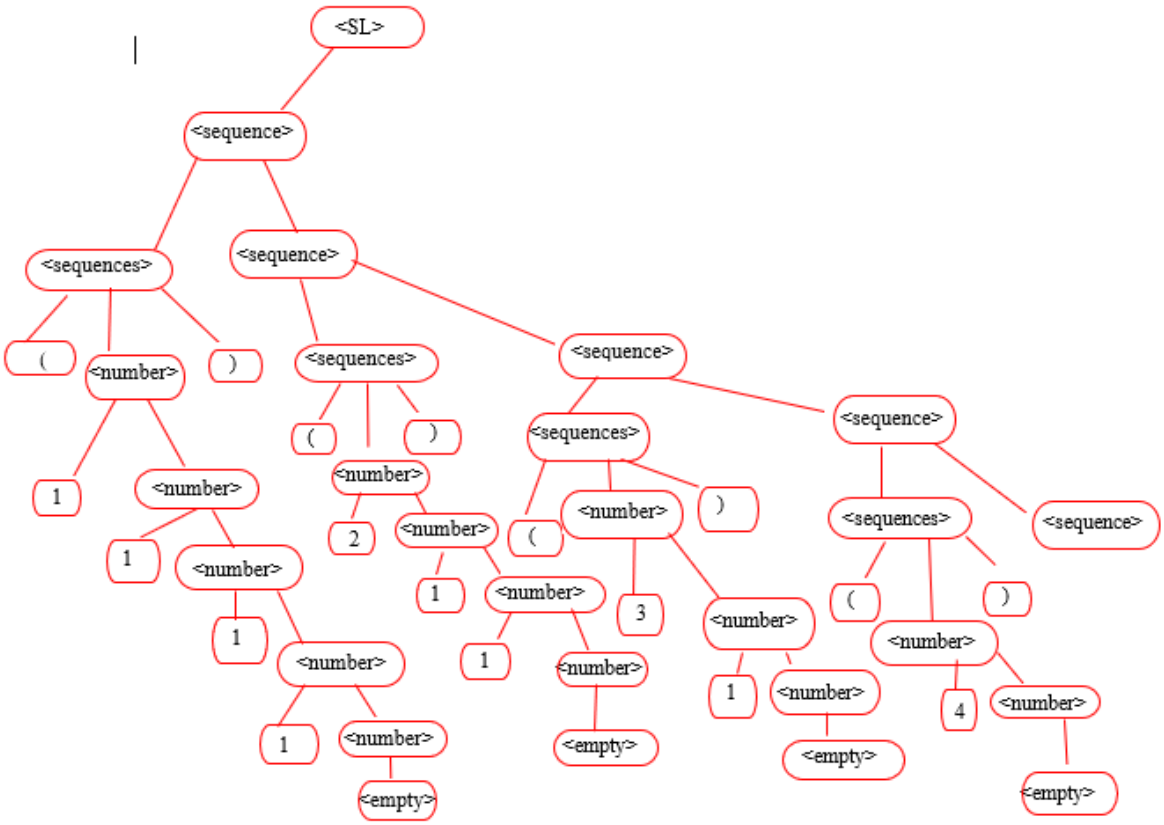
2. Parse Tree for: (4)



3. Parse Tree for: (1 1 1 1) (3 1) (1 3) (3 1)



4. Parse Tree for: (1 1 1 1) (2 1 1) (3 1) (4)

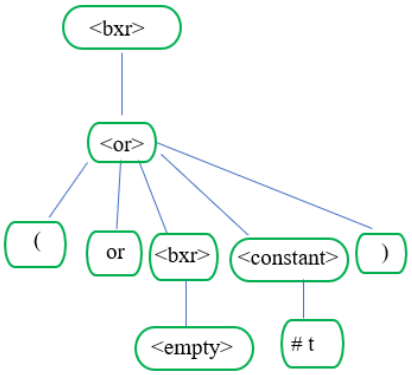


Problem 4 - BXR

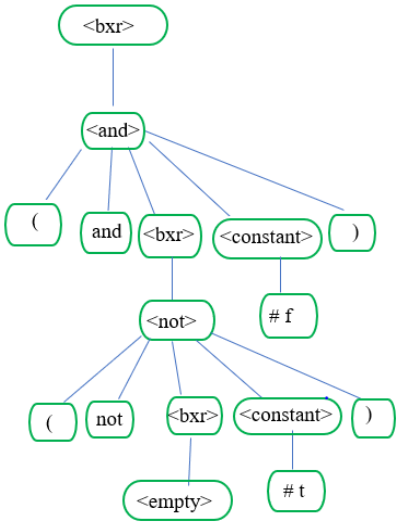
1. **Grammar Description of BXR**

<bxr> : : = <empty> | <and> | <or> | <not>
<and> : : = (and <bxr> <constant>)
<or> : : = (or <bxr> <constant>)
<not> : : = (not <bxr> <constant>)
<constant> : : = #t | #f

2. **Parse Tree for: (or #t)**



3. **Parse Tree for: (and (not #t) # f)**

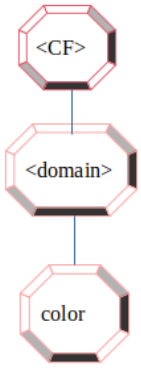


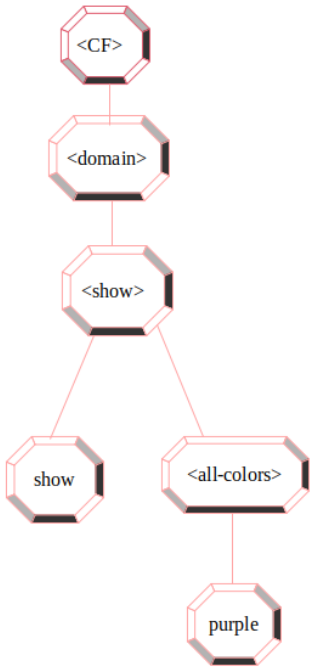
Problem 5 - CF (Color Fun)

1. **Grammar Description for: CF (Color Fun)**

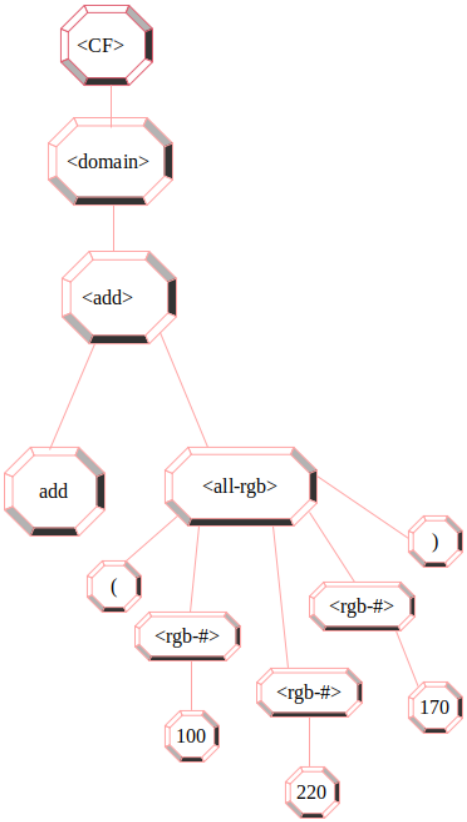
<CF> : : = <domain>
<domain> : : = color | <empty> | <add> | <show>
<add> : : = (add color < all-colors>) | (add <all-rgb>)
<show> : : = show <all-color>
<all-rgb> : : = (<rgb-#> <rgb-#> <rgb-#>)

2. **Parse tree for : color and show purple**





3. **Parse Tree for: add (100 220 170)**



Problem 6 - BNF?

BNF (Backus-Naur Form) is a context-free grammar often used by computer language developers to establish a language's syntax rules. John Backus was a programmer who created a notation to document IAL (an early implementation of Algol). Syntactically correct sentences are those that are derived utilizing the production rules. A sentence's syntax may be checked by creating a parse tree that shows how the sentence is produced from the production rules. If such a tree cannot be constructed, the phrase has syntactic mistakes.