Review Set 2

Kevin Leach

September 21, 2023

1 Cool Syntax

1. Consider the Cool program below

```
1  class Main inherits IO {
2      main () : Object {
3          out_string((new Language).________________________________________)
4      };
5  };
6
7  class Language {
8      newLanguage (x:Int) : Language {
9          if x = 0 then
10             new English
11         else
12         (* complete this *)
13
14
15
16
17
18         fi
19     };
20     greeting () : String {""};
21 };
22
23 class English inherits Language {
24     greeting () : String {"hello"};
25 };
26
27 class Spanish inherits Language {
28     greeting () : String {"hola"};
29 };
30
31 class Chinese inherits Language {
32     greeting () : String {"nihao"};
33 };
```

2. Complete the `newLanguage` method so that it returns a new `Spanish` object if x is 1 and a new `Chinese` object otherwise.

3. Complete the `main` method so that it uses the `greeting()` method to output "hello" if given 0 from stdin, "hola" if 1, and "nihao" otherwise.

2 Regular Expressions

1. Write a regular expression over the alphabet $\Sigma = \{a, b, c, d\}$ for the language of strings that have a $1 + 3n$ occurrences of a followed by any combination of even numbers of occurrences of b or c, followed by one or more occurrences of d.

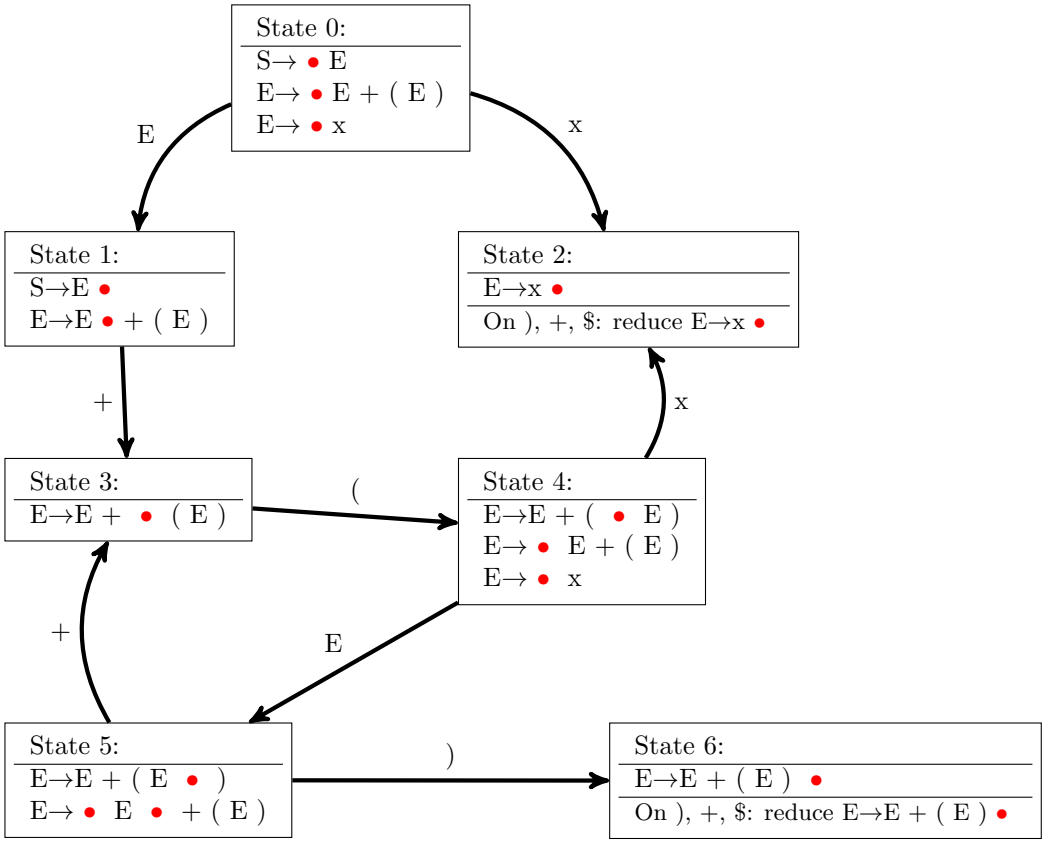
2. Draw a **DFA** that accepts the language from the above problem.

3 Context-Free Grammars and Parsing

1. ALWAYS, SOMETIMES, NEVER. If a grammar g cannot be LL(1) parsed, then $L(g)$ is not regular.

2. ALWAYS, SOMETIMES, NEVER. If a recursive descent parser can parse a grammar g, the $L(G)$ is LL(1).

A LR parsing DFA is shown below.



3. In the table below, show the parsing steps for the string `x + ( x )`

Stack	**Action**
• x + (x)$	Shift x
x • + (x)$	Reduce $E \rightarrow$x
• E + (x)$	shift E

Stack	**Action**