Lesson 1
Compiler Design I (Kompilatorteknik I) 2016

1 Lexical Analysis

Let $L$ be a language of strings on $\Sigma = \{x, y\}$ defined by $L = \{x, xxx, (xxxx)^*y\}$, that is $\{x, xxx\} \cup \{x^n y | n \geq 0\}$.

1. Construct a DFA that accepts $L$. (Hint: Begin with an NFA and then convert it)

   Answer:

   The states have been labeled either with the token that would be emitted (accepting) or with a symbolic name corresponding to the input that has been read so far (non-accepting). The error state has been omitted.

2. Describe how the lexical analyzer will tokenize the following inputs:

   • $xx$
     
     Answer:

     Reading two ’$x$’s, the lexical analyzer reaches state “$xx$”. The input ends in a non-accepting state, so the analyzer backtracks to the most recent accepting state (“$x$”) and outputs the corresponding token, consuming just one of the two ’$x$’s. Starting from scratch and reading the remaining ’$x$’ again, the analyzer reaches the “$x$” state again, and the input ends there so it emits a second “$x$” token.

   • $xxxxxxxxy$
     
     Answer:

     With the given input, the DFA reaches the “$(xxxx)^*y$” state. The input ends there, so the analyzer emits just one token of type “$(xxxx)^*y$".
Answer:

The given input drives the DFA to the \((xxx)^*x\) state before the 'y' is read, leading to the null state. The analyzer backtracks, therefore, and emits the most recent acceptable token: “xxx”, leaving the rest 10 'x's to be scanned again. Starting from scratch, the state \((xxx)^*xx\) is reached before 'y' is read, so the analyzer backtracks again, emitting a second “xxx” token and leaving 7 'x's to be re-scanned. A third pass will reach state \((xxx)^*xxx\) before 'y' again leads to an error, so a third “xxx” token is emitted after backtracking, with 4 'x's remaining. Finally the rest input leads to the accepting state \((xxx)^*y\), which is the last token that the analyzer will emit.