# Informatics 2A: Tutorial Sheet 3 - SOLUTIONS

John Longley

1. Suppose $a \in \Sigma$.

   (a) The string $aaa$ can be derived in two ways:
   This ambiguity is harmless since both trees define the language $\{aaa\}$.

   (b) The string $aa*$ can also be derived in two ways.
   This ambiguity is harmful since one parse tree defines the regular language $\{a^n \mid n \geq 1\}$ and the other defines $\{a^{2n} \mid n \geq 0\}$. Note that the first parsing is the one that respects the usual precedence conventions.

2. (a) The idea is that we jump to the second state once the b's start coming. Formally, the control states are $Q = \{q_a, q_b\}$, the stack alphabet is $\Gamma = \{\perp\}$, and we include the following transitions:

$$\begin{array}{ccc} q_a & \xrightarrow{\$,\perp:\epsilon} & q_a \\ q_a & \xrightarrow{a,\perp:\perp\perp} & q_a \\ q_a & \xrightarrow{b,\perp:\epsilon} & q_b \\ q_b & \xrightarrow{b,\perp:\epsilon} & q_b \\ q_b & \xrightarrow{\$,\perp:\epsilon} & q_b \end{array}$$

   (b) Let $\Gamma = \{(, [, \perp\}$. Consider a single state with the following self-transitions.

$$\begin{array}{ll} (, x : (x & \text{for each } x \in \Gamma \\ [, x : [x & \text{for each } x \in \Gamma \\ ), ( : \epsilon & \\ ], [ : \epsilon & \\ \$, \perp : \epsilon & \end{array}$$

3. (a)

| Operation | Input remaining | Stack state |
|---|---|---|
| | ( n * n )$ | Exp |
| Lookup (,Exp | ( n * n )$ | ( Exp ) |
| Match ( | n * n )$ | Exp ) |
| Lookup n, Exp | n * n )$ | n Ops ) |
| Match n | * n )$ | Ops ) |
| Lookup *, Ops | * n )$ | * n Ops ) |
| Match * | n )$ | n Ops ) |
| Match n | )$ | Ops ) |
| Lookup ), Ops | )$ | ) |
| Match ) | $ | STACK EMPTIES AT END OF STRING: SUCCESS! |

   (b)
   - For (), the parser will encounter a blank table entry at ), Exp. Message: ") Found where expression expected."

- For n), the stack will empty before end of input is reached.
  Message: ") Found after end of expression."
- For n*, the end of input will be reached with n Ops still on the stack, and the parser gets stuck since the top of the stack is a terminal n no different from $.
  Message: "End of input found where numeric literal expected."

(c)

$$\begin{aligned} \mathsf{Exp} &\rightarrow \mathsf{ExpA\,Ops} \\ \mathsf{Ops} &\rightarrow \epsilon \mid * \,\mathsf{ExpA\ Ops} \\ \mathsf{ExpA} &\rightarrow \mathsf{n} \mid (\,\mathsf{Exp}\,) \end{aligned}$$

(Other solutions are possible.)

4. (a) $\mathsf{E} = \{\mathsf{OptMinus},\ \mathsf{TimesOps},\ \mathsf{PlusOps}\}$

   (b) $First(\mathsf{OptMinus}) = \{-, \epsilon\}$
   $First(\mathsf{TimesOps}) = \{*, \epsilon\}$
   $First(\mathsf{PlusOps}) = \{+, \epsilon\}$
   $First(\mathsf{Exp}) = First(\mathsf{Cond}) = First(\mathsf{TimesExp}) = \{-,\mathsf{n}\}$

   (c) $Follow(\mathsf{Cond}) = \{\$\}$
   $Follow(\mathsf{Exp}) = \{\$, ==\}$
   $Follow(\mathsf{PlusOps}) = \{\$, ==\}$
   $Follow(\mathsf{TimesExp}) = \{\$, ==, +\}$
   $Follow(\mathsf{TimesOps}) = \{\$, ==, +\}$
   $Follow(\mathsf{OptMinus}) = \{\mathsf{n}\}$

   (d) The grammar is indeed LL(1)!
   Parse table as follows.

| | n | + | * | - | == | $ |
|---|---|---|---|---|---|---|
| Cond | Exp==Exp | | | Exp==Exp | | |
| Exp | TimesExp PlusOps | | | TimesExp PlusOps | | |
| TimesExp | OptMinus n TimesOps | | | OptMinus n TimesOps | | |
| OptMinus | $\epsilon$ | | | - | | |
| TimesOps | | $\epsilon$ | * n TimesOps | | $\epsilon$ | $\epsilon$ |
| PlusOps | | + TimesExp PlusOps | | | $\epsilon$ | $\epsilon$ |