

# 11-711 – Algorithms for NLP

## Midterm Sample Solutions

November 8, 2002

### 1

#### 1.1

$S \rightarrow 0S1$

$S \rightarrow 1S0$

$S \rightarrow \epsilon$

#### 1.2

$S \rightarrow AB$

$B \rightarrow SC$

$S \rightarrow AC$

$S \rightarrow CD$

$D \rightarrow SA$

$S \rightarrow CA$

$A \rightarrow 0$

$C \rightarrow 1$

#### 1.3

$Q = q, \Sigma = 0, 1, \Gamma = \Sigma \cup S, A, B, C, D, q_0 = q, z_0 = S, F = \phi$

$\delta$  is defined as below:

[1]  $\delta(q, \epsilon, S) = \{(q, 0S1), (q, 1S0), (q, \epsilon)\}$

[2]  $\delta(q, 1, 1) = (q, \epsilon)$

[3]  $\delta(q, 0, 0) = (q, \epsilon)$

This is a PDA which will replace a non-terminal from  $\Gamma$  on the top of the stack with an equivalent string of non-terminal and/or terminals from  $\Sigma$  (rule [1]). Also, if the next terminal to be consumed matches the terminal on the top of the stack, it is consumed and the terminal is popped off the stack (rules [2] and [3]). If the input word is consumed completely and there is a valid derivation, the stack will be empty and the PDA will accept the word. Otherwise, if no derivation matches the word, the PDA will not be able to empty the stack and thus reject.

## 2

$(q_0, 011001, z_0) = (q, 011001, S) \vdash$

$(q, 011001, 0S1) \vdash (q, 11001, S1) \vdash (q, 11001, 1S01) \vdash (q, 1001, S01) \vdash (q, 1001, 1S001) \vdash$   
 $(q, 001, S001) \vdash (q, 001, 001) \vdash (1, 01, 01) \vdash (q, 1, 1)$

$\vdash (q, \epsilon, \epsilon)$

## 3

Let  $n$  be the constant from the PL for this language. Let  $w = 0^n 1^n$ . Then  $rev(w) = 1^n 0^n = \bar{w}$  so  $w \in L$ . Now let us consider the decomposition of  $w$  according to the PL:  $w = xyz$  such that  $|xy| \leq n$ ,  $|y| \geq 1$ . Then  $x$  and  $y$  are both strings of 0's since  $w$  starts with  $n$  0's. Let  $p = |y|$  and  $|x| = k$ , so  $z = 0^{n-(p+k)} 1^n$ . Now, for instance, with  $i = 0$ , the word  $xy^i z = xz = 0^k 0^{n-(p+k)} 1^n = 0^{n-p} 1^n$  with  $p \geq 1$ . But  $rev(xz) = 1^n 0^{n-p}$  and  $\bar{xz} = 1^{n-p} 0^n$  with  $p \neq 0$  so  $xz \notin L$ , for any decomposition  $x, y, z$  such that  $|xy| \leq n, |y| \geq 1$ . Hence, the PL does not hold and  $L$  is not regular.

## 4

### 4.1

See figures 1 and 2.

### 4.2

(Please note that the following is only one possible acceptable h'.)

Let  $h' = 2n - k - 1$  where  $n$  is the number of words in the sentence and  $k$  is the number of steps taken to derive the current node.

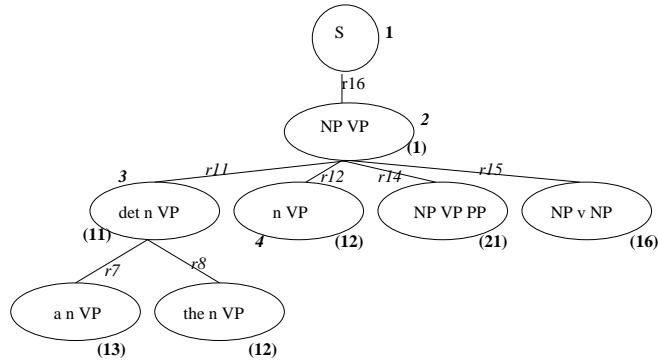


Figure 1: Uniform Cost Search

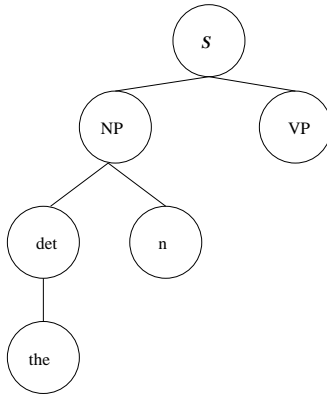


Figure 2: Parse tree for last state produced

This heuristic is admissible because the grammar is basically in CNF form with the exception of rule  $r12$ . We can change the grammar to CNF by replacing  $r12$  with  $NP \rightarrow John|boy|saw|telescope$ . Thus, the presence of  $r12$  will only increase the number of derivation steps for input of length  $n$  in a CNF grammar, i.e.  $2n - 1$ . Thus, the number of steps to derive a sentence in this grammar is  $\geq 2n - 1$ . At each node,  $k$  steps have already been taken so  $\geq 2n - k - 1$  more steps are needed. Since the minimum cost of each step is 1,  $h'$  always underestimates  $h$  and is therefore admissible.

For the A\* search, see figure 3. The parse tree for the last state produced is the same as in the previous problem (figure 2).

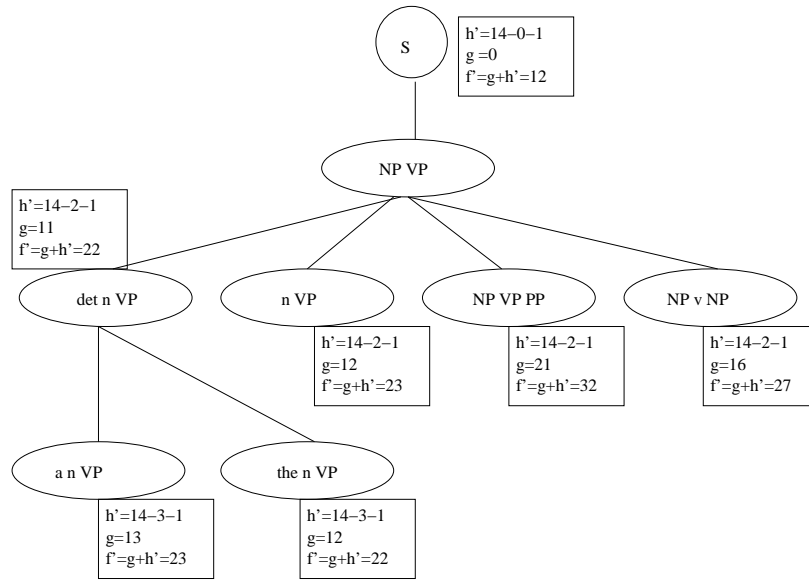


Figure 3: A\* search

### 4.3 (extra credit)

No, it is not possible to give an upper bound on the number of steps. The search will go into an infinite loop when applying the rule  $VP \rightarrow VP PP$ . This is because this rule ( $r_{14}$ ) is a left-recursive rule and we are expanding by a leftmost derivation.