Michael Wang
CS 181 HW 5

0a) Two other ways to show that a grammar is ambiguous are showing multiple left-most derivations for the same string or multiple right-most derivations for the same string.

0b) Using the left-most derivation for the string (()) we have:

S → (S) →((S)) → (()) OR

S → (S) →(SS) → (SSS) →(()SS) → (()S) → (())

1) A known non-FSL is {$a^n b^n$}. This is clearly a proper subset of the language {a*b*} which being a regular expression is a finite state language.

2) FSL are closed under complement. L' is an FSL if and only if L is an FSL. Since L' is not an FSL, neither is L.

3) Assume the language L is an FSL. Then there exists a pumping length p. Consider string s = $a^p b^{p+p!}$. |s| > p and s is a member of L so we must be able to split s into xyz such that $xy^i z$ is in L for all i greater than 0. Since |xy| <= p, y must be all a's, i.e. y = a^n for some number less than p. Thus pumping s gives us a $a^{p+ni} b^{p+p!}$. However, all numbers less than p are a divisor of p! so no matter how we choose y p+ni = p+p! for some i and the string is excluded from L. Thus we have a contradiction and L cannot be a FSL.