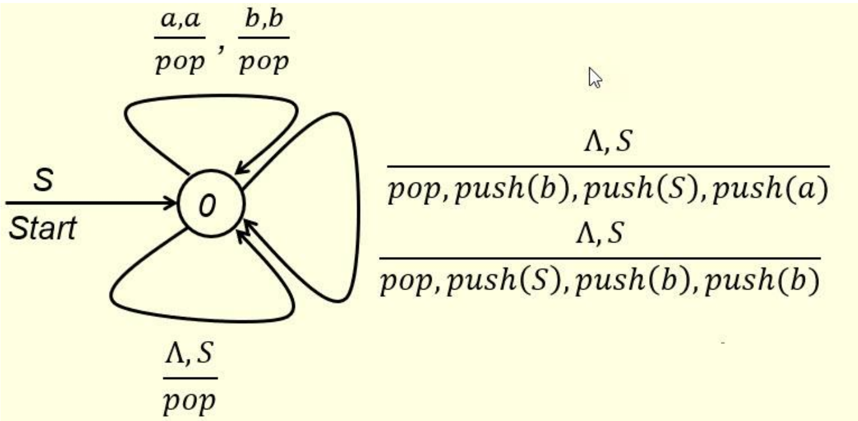
CS375 Homework Assignment 6 Solution Set (40 points)

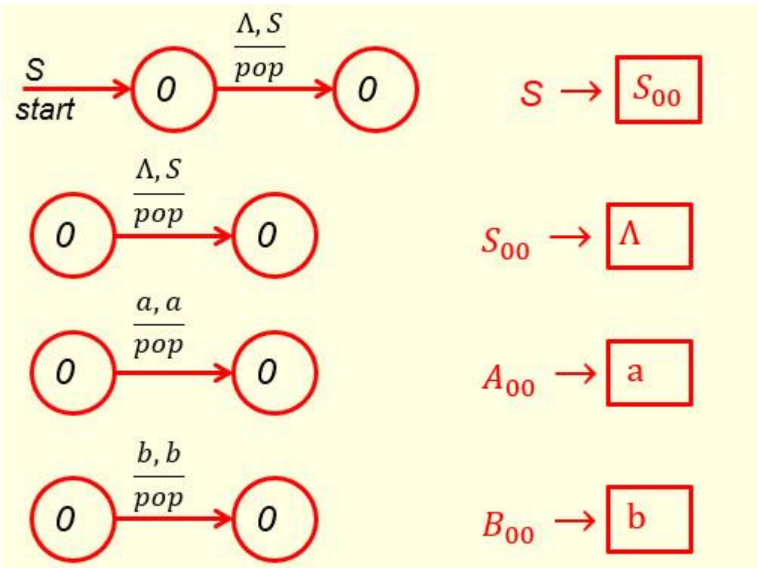
Due date: 10/30/2024

1. (6 points)

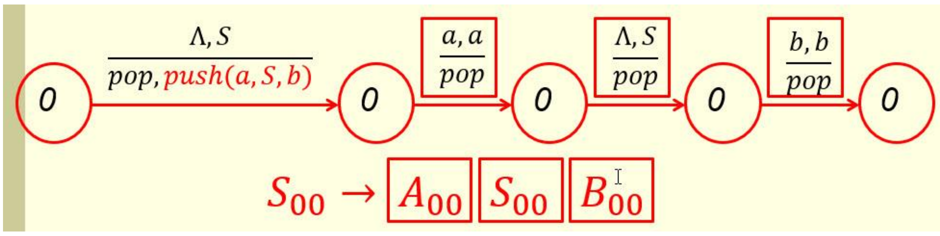
   Given the context-free grammar {S → Λ ; S → aSb ; S → bbS }, we can convert it to a one-state empty-stack acceptance PDA as follows.

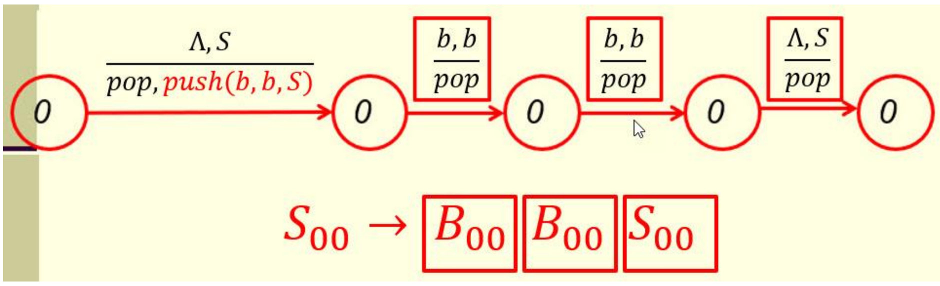


On the other hand, given such a one-state empty-stack acceptance PDA, we can convert it to a CFG. In this case, we have one type-4 path, three type-1 paths and two general type-3 paths. The type-4 and type-1 paths and their corresponding CFG productions are shown below.



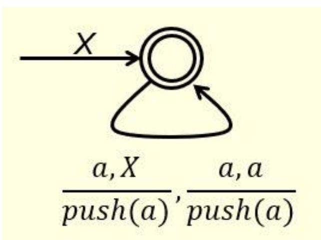
In the following, fill out the blanks in the general type-3 paths for $\frac{\Lambda, S}{pop, push(b), push(S), push(a)}$ and $\frac{\Lambda, S}{pop, push(S), push(b), push(b)}$ and the blanks in the corresponding CFG productions.





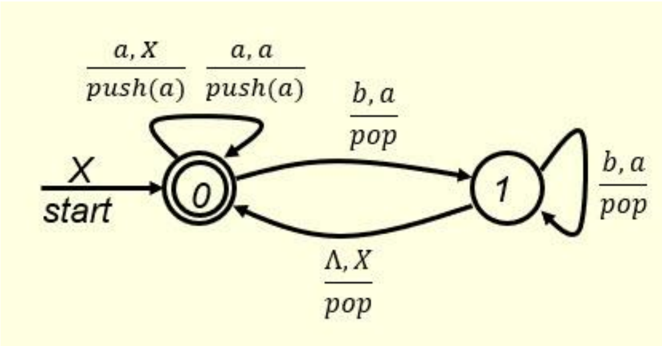
After a simple simplification process, we would get a CFG exactly the same as the given one.

2. (1 point)
Final-state acceptance and empty-stack acceptance are equivalent only for NPDA's. They are not equivalent for DPDA's. For DPDA's the class of languages defined by final-state acceptance is bigger. In the following, use the given DPDA



to show that this is indeed the case by pointing out the language L = { a^n | n ϵ **N** } is accepted by the given (final-state) DPDA, but is not accepted by the DPDA when viewed as an empty-stack DPDA.

3. (5 points)
Given the following final-state DPDA,



and the following strings

Λ, aa, bb, aaa, bbb, ab, ba, aabb, bbaa, aaabbb, bbbaaa

which of these strings are accepted by the given final-state DPDA? Put your answer in the following blank.

Λ, aa, aaa, ab, aabb, aaabbb

(1.5 points)

If the given final-state DPDA is considered as an empty-stack NPDA (state 0 is no longer a final state), then which of the given strings are accepted by the empty-stack DPDA? Put your answer in the following blank.

ab, aabb, aaabbb

(1.5 points)

Now, consider the following two general questions. First, what is the language L_1 accepted by the given final-state DPDA? Put your answer in the following blank.

$L_1 = \{ a^n, a^m b^m \mid n \in N; m \in N \}$

or

$L_1 = \{ a^n, a^m b^m \mid n \in N; m \in N^+ \}$

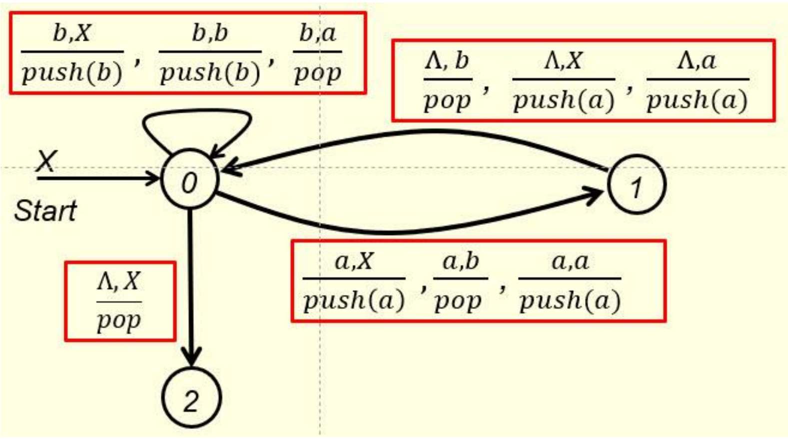
(1 point)

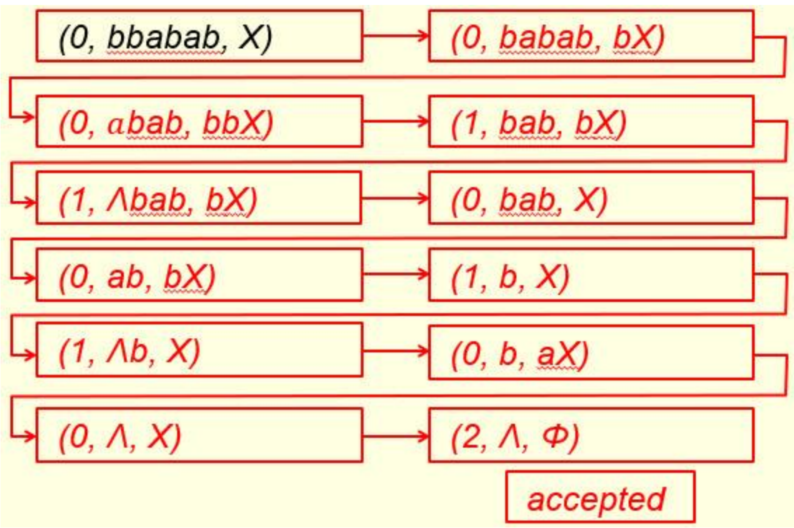
Second, what is the language L_2 accepted by this DPDA when viewed as an empty-stack DPDA? Put your answer in the following blank.

$$L_2 = \{\, a^m b^m \mid m \in N^+ \,\}$$ (1 point)

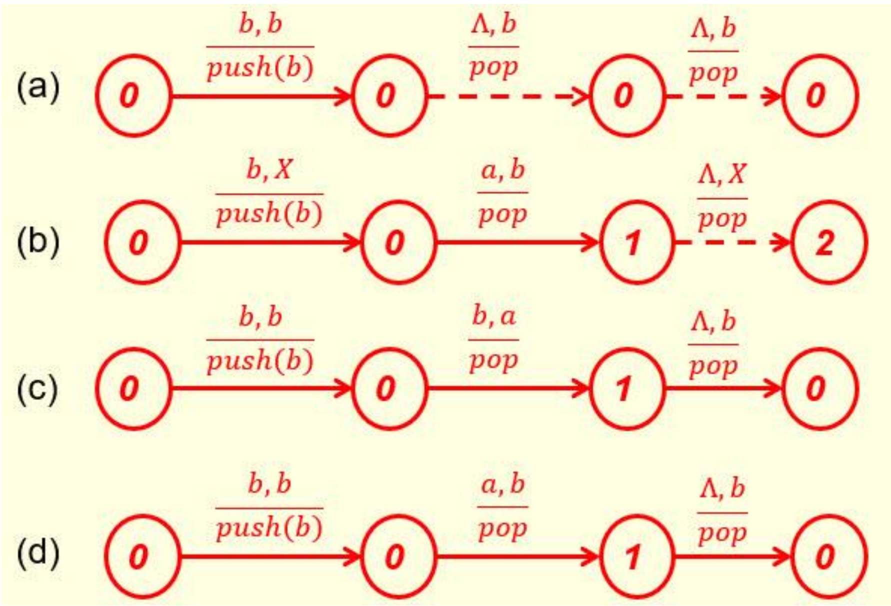
L_1 obviously is bigger than L_2.

4. (5 points)
The following empty-stack PDA accepts the language L = { w ϵ {a, b}* | n_b(w) = 2n_a(w) } (assuming $\Lambda \in L$). In the following blanks show the execution of the string bbabab by this PDA.





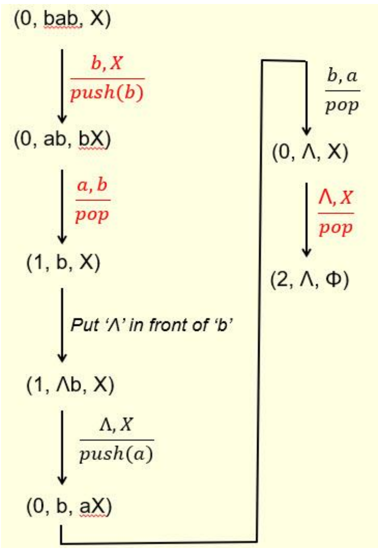
5. (4 points)
The empty-stack PDA given in question #4 has one type 4, four type 1 and eight type 3 instructions. In the following four possible type 3 instructions, which one(s) are legitimate type 3 (i.e., they really exist)?



Put your answer in the following blank.

(b), (d)

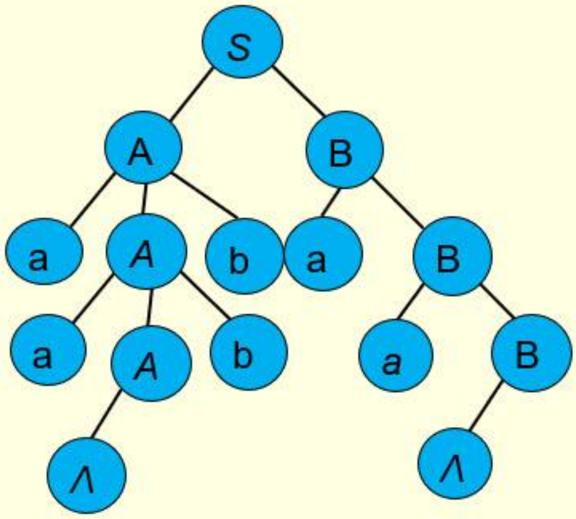
For case (b), consider the string 'bab' which is a member of the language L accepted by the PDA given in Question #4.
The execution of 'bab' by the PDA given in Question #4 can be performed as follows:



So, by combining the first, the second and the last instructions we get a type 3 path.

6. (6 points)

Given the following parse tree where S, A, B are non-terminals, a and b are terminals and Λ is the empty string, show the corresponding left-most derivation of the yield in the blanks on the right side. (6 points)



S ⇒ AB

⇒ aAbB

⇒ aaAbbB

⇒ aaΛbbB

⇒ aaΛbbaB

⇒ aaΛbbaaB

⇒ aaΛbbaaΛ

⇒ aabbaa

(0.5 points each)

It is easy to see that the grammar used to build the above parse tree has the following productions:

S → AB A → aAb | Λ B → aB | Λ

The language of this grammar is:

$$\{a^n b^n a^m \mid n, m \in N\}$$

Does the derivation show the grammar is an LL(1) grammar?

[ ] Yes [x] No (1 point)

(Hint: consider the input string 'aa')

Does the derivation show the grammar is an LL(2) grammar?

[ ] Yes [x] No (1 point)

(Hint: consider the same input string 'aa')

7. (2 points)

I claim the following grammar for $\{a^{m+n}b^{m}c^{n} \mid m,n \in N\}$ is an LL(1) grammar.

S → aSc | T   T → aTb | Λ

My justification is that I can build a leftmost derivation for the string aaabcc by examining only one input symbol for each step of the derivation. The leftmost derivation is shown below.

S ⇒	aSc	(step 1)
⇒	aaScc	(step 2)
⇒	aaTcc	(step 3)
⇒	aaaTbcc	(step 4)
⇒	aaabcc	(step 5)

If you think the above derivation is correct, mark the True box below. Otherwise, mark the False box and give your reason in the box below the correct box.

True [ ]   *False* [X]

What is wrong with the derivation:

Step 3 is not correct.
According to the third symbol in the input string, the production S → aSc should be selected, not S → T.

8. (4 points)

Given the following context-free grammars for the language $\{a^{m+n}b^{m}c^{n} \mid m,n \in N\}$,

(a) S → aSc | aBb | Λ
B → aBb | Λ

(b) S → aSc | B | Λ
B → aBb | Λ

(c) S → aSc | B
B → aBb | Λ

(i) which one or ones are LL(1)? (2 points)

None.
Hint: try ‘ab’ as input string for all three cases.

(ii) which one or ones are ambiguous? (2 points)

None.
Note that the production ‘S → aSc’ will always be used before any ‘S → aBb’ can be used in the construction of the parse tree for an input string, except when the input string contains no c’s. In such a case, only ‘S → aBb’ will be used in the construction of the parse tree. What this means is: the parse tree of each input string is of a unique structure. Hence, cannot be ambiguous.

9. (4 points)
Given the following context-free grammars for the language $\{a^{m+n}b^{m}c^{n}$ | m,n ϵ N}, which one or ones are LL(2) but not LL(1)?

(a) S → aaScc | aaBbc | aaBbb | aBb | ac | Λ
    B → aBb | Λ
(b) S → aaScc | aaBbc | aBb | ac | Λ
    B → aBb | Λ
(c) S → aaScc | aaBbc | B | ac | Λ
    B → aBb | Λ
(d) S → aaScc | aaBbc | B | ac
    B → aBb | Λ

None.
Hint: try ‘aabb’ as input string for all four cases.

10. (3 points)

The language generated by the following grammar is (1 point)

$\{a^m(ab)^n \mid m, n \in N\}$

S → aS | A | Λ   A → abA | Λ

Is this an LL(1) grammar?	[ ] Yes	[x] No	(1 point)
Is this an LL(2) grammar?	[ ] Yes	[x] No	(1 point)

Hint: consider 'aab' as input string in both cases.