COL352 HW-3

Sarthak Agrawal 2018CS10383
Piyush Gupta 2018CS10365
Navneel Singhal 2018CS10360

April 2, 2021

Problem 1

Consider the grammar $G = (\{S\}, \{a, b\}, R, S)$, where the set of production rules is

$$R = \{S \longrightarrow SS,\ S \longrightarrow aaSb,\ S \longrightarrow bSaa,\ S \longrightarrow aSbSa,\ S \longrightarrow \varepsilon\}.$$

Obviously, the language L generated by this grammar contains only those strings in which the number of a's is twice the number of b's. (If you are unable to prove the last statement rigorously, you must contact the instructor asap.) Prove that, in fact, L contains all strings in which the number of a's is twice the number of b's.

Solution.

Notation 1

Let us define $C(s)$ for $s \in \{a, b\}^*$ to be equal to $\#a's - 2\#b's$. Let's call a string $s \in \{a, b\}^*$ **balanced** if $C(s) = 0$.

Lemma 1.1

Let s be a non-empty balanced string. If b is the first character of s then either aa is a suffix of s, or, s can be partitioned into 2 (non-empty) substrings s_1, s_2 such that $s = s_1 s_2$ and both s_1, s_2 are balanced.

Proof. Firstly, since s is balanced, its length must be three times the number of bs in it, so it is at least 3 characters long. We shall suppose that s can not be partitioned into 2 balanced strings and then show that s must have aa as its suffix using proof by contradiction.

Suppose s also does not end in aa. Let length of s be n. Now, note that

1. $C(s[1 \dots 1]) = C(b) = -2$
2. $C(s[1 \dots n-2]) = -C(s[n-1 \dots n])$ and hence > 0 (since there is at least one b in the last two characters by assumption).

Thus, if we consider the sequence of values $C(s[1 \dots i])$ for $i \in [1, n-2]$ we shall find that it starts with an initial value of -2, and terminates at a positive value. Since by definition the only possible positive increase in the $C-$value is 1, it must be the case that at some point C value must have become zero. In other words, $C(s[1 \dots k]) = 0$ for some $1 < k < n-2$. But this is a contradiction because then we can partition s into 2 substrings $s[1 \dots k]$ and $s[k+1 \dots n]$ both of which must be balanced. Hence our supposition that s does not end in aa must be incorrect. □

Corollary 1

Let s be a non-empty balanced string. If s ends with b then either aa is a prefix of s, or, s can be partitioned into 2 (non-empty) substrings s_1, s_2 such that $s = s_1 s_2$ and both s_1, s_2 are balanced.

Proof. The corollary follows from the lemma above by observing that if s is balanced then so is $rev(s)$, and since then b becomes the first character of $rev(s)$ the corollary follows. □

Now, we need to show that if s is a **balanced** string then $S \overset{*}{\Longrightarrow} s$. We shall show by induction on length of s.

Base Case $|s| = 0$. That is, $s = \varepsilon$. Then by rule 5 we have $S \Longrightarrow s$

Inductive Case We will do case analysis on string s

1. **Case 1**. s can be partitioned into 2 (non-empty) substrings s_1, s_2 such that $s = s_1 s_2$ and both s_1, s_2 are balanced. By inductive hypothesis we must have $S \overset{*}{\Longrightarrow} s_1$ and $S \overset{*}{\Longrightarrow} s_2$. Thus, we can write $SS \overset{*}{\Longrightarrow} s_1 s_2$, or in other words, $SS \overset{*}{\Longrightarrow} s$. But by rule 1 we have $S \Longrightarrow SS$. Therefore we have that $S \Longrightarrow SS \overset{*}{\Longrightarrow} s$.

2. **Case 2**. s does not belong to above case and s begins with character b. Then, by our lemma we know that s must end in suffix aa. Thus, s must be of form $bs'aa$ and clearly s' is also a balanced string whose length is less than $|s|$. So by induction hypothesis $S \overset{*}{\Longrightarrow} s'$ which gives $bSaa \overset{*}{\Longrightarrow} bs'aa$. Combined with rule 3 this results in $S \Longrightarrow bSaa \overset{*}{\Longrightarrow} bs'aa = s$.

3. **Case 3**. s does not belong to above case(s) and s ends in character b. Then, by our corollary to the lemma we know that s must begin with prefix aa. Thus, s must be of form $aas'b$ and clearly s' is also a balanced string whose length is less than $|s|$. So by induction hypothesis $S \overset{*}{\Longrightarrow} s'$ which gives $aaSb \overset{*}{\Longrightarrow} aas'b$. Combined with rule 2 this results in $S \Longrightarrow aaSb \overset{*}{\Longrightarrow} aas'b = s$.

4. **Case 4**. s does not belong to any of the above cases. This means that s must begin with, and end in character a as well as s can not be suitably partitioned into balanced strings. So, let s be of form $as'a$. Then, note that because s is balanced $C(s) = 0$ which implies $C(as') = -1$. Further, $C(a) = 1$. Note that $C(.)$ value of a prefix of s can only decrease by 2 at a time (and that too at an occurrence of b), and it transitioned from a positive value for $C(s[1 \dots 1])$ to a non-negative value for $C(s[1 \dots n-1])$, so there must exist an index $i > 1$ such that $s[i] = b$ and $C(s[1 \dots i-1]) = 1$ (the other case with the transition from a balance of 2 to a balance of 0 is impossible, since it would contradict the fact that s can't be partitioned into two balanced substrings).

   Hence the substrings $s[2 \dots i-1]$ and $s[i+1 \dots n-1]$ must be balanced ($C(.) = 0$). So we have reduced s into the form $s = as_1bs_2a$ where s_1 and s_2 are balanced. By induction, $S \overset{*}{\Longrightarrow} s_1$ and $S \overset{*}{\Longrightarrow} s_2$. Further, $S \Longrightarrow aSbSa$ by rule 4. So together we have $S \Longrightarrow aSbSa \overset{*}{\Longrightarrow} as_1bs_2a = s$.

This completes our inductive step, establishing the inductive hypothesis.

By the inductive hypothesis we have proved that all balanced strings are generated by the given grammar.

Problem 2

An *empty stack pushdown automaton* is a tuple $P' = (Q, \Sigma, \Gamma, \Delta, q_0, Z)$ where Q, Σ, Γ, Δ, and q_0 have the same meaning as in definition of a PDA. The last component $Z \in \Gamma$ is the initial stack symbol: it is present on the stack in the beginning of every run. In other words, the initial instantaneous description in a run of P' on $x \in \Sigma^*$ is (q_0, x, Z). We say that such an automaton P' accepts string x if there is a run of P' on x that ends with an empty stack (and in any state). Formally, P' accepts x if there exists a state $q \in Q$ such that $(q_0, x, Z) \vdash_{P'}^* (q, \varepsilon, \varepsilon)$. The language recognized by an empty stack pushdown automaton P' is the set of strings accepted by P'.

1. **[1 mark]** Prove that if a language L is recognized by some PDA P, then it is recognized by some empty stack PDA P'. (Describe a construction of P' from P.)
2. **[1 mark]** Prove that if a language L is recognized by some empty stack PDA P', then it is recognized by some PDA P. (Describe a construction of P from P'.)

Solution.

1. Consider any PDA $P = (Q, \Sigma, \Gamma, \Delta, q_0, A)$. We shall construct an empty stack PDA $P' = (Q', \Sigma, \Gamma', \Delta', q_{init}, A, Z)$ as follows:

1. $Q' = Q \uplus \{q_{init}, q_{accept}\}$.
2. $\Gamma' = \Gamma \uplus \{\bot, Z\}$.
3. $\Delta' = \Delta \cup \{(q_{init}, \epsilon, Z, q_0, \bot)\} \cup \{(q, \epsilon, \epsilon, q_{accept}, \epsilon) \mid q \in A\} \cup \{(q_{accept}, \epsilon, a, q_{accept}, \epsilon) \mid a \in \Gamma'\}$.

We prove the claim in the problem in two parts.

Claim 2.1

If P accepts a string x, then P' accepts x.

Proof. Consider an "accepting" run of P on x. More formally, we have $(q_0, x, \epsilon) \vdash_P^* (q, \epsilon, s')$ for some $q \in A$ and $s' \in \Gamma^*$. Note that this implies that we have $(q_0, x, \bot) \vdash_P^* (q, \epsilon, s'\bot)$, and since $\Delta \subseteq \Delta'$, we have $(q_0, x, \bot) \vdash_{P'}^* (q, \epsilon, s'\bot)$. By the definition of Δ', we have $(q_{init}, x, Z) \vdash_{P'} (q_0, x, \bot) \vdash_{P'}^* (q, \epsilon, s)$ for $s = s'\bot \in \Gamma'^*$ and $q \in A$. From the third set in Δ', we get that $(q, \epsilon, s) \vdash_{P'} (q_{accept}, \epsilon, s)$, and since the last set represents transitions that lead to q_{accept} for any stack character, we have $(q_{accept}, \epsilon, s) \vdash_{P'}^* (q_{accept}, \epsilon, \epsilon)$. Combining this with the previous relation, we get $(q_{init}, x, Z) \vdash_{P'}^* (q_{accept}, \epsilon, \epsilon)$, as needed. □

Claim 2.2

If P' accepts a string x, then P accepts x.

Proof. Consider an "accepting run" of P' on x. We call an i.d. *good* if it is of the form (q, ϵ, ϵ) for some $q \in Q'$. An accepting run can end only at a good i.d..

The first i.d. in the run must be (q_{init}, x, Z) (which is not good), and the second one must exist and be equal to $(q_0, x, \bot)$, which is also not good.

We first claim that the last i.d. in the run is $(q_{accept}, \epsilon, \epsilon)$. Note that since the run is accepting, the last state has to be good. Since the second i.d. in the run has $\bot$ on the stack, and the last doesn't, there has to be some transition in the run where $\bot$ is popped. This must be done in a transition from q_{accept} to itself, since no other transition involves popping of $\bot$. Moreover, this transition is the only such transition, since we push $\bot$ in only the first transition, and no other transition is able to do so.

Hence, in the run, we reach q_{accept} at some point, and since any transition out of q_{accept} leads to q_{accept}, the last state in the run is q_{accept}.

Consider the last i.d. in the run that is not at state q_{accept}. This is clearly not the last i.d. in the run itself, since the the last i.d. in the run is at state q_{accept}. Note that the part of the run from this i.d. to the final i.d. doesn't involve reading and discarding characters from the string at that point, so

this i.d. is of the form (q, ϵ, s) for some $q \in Q, s \in \Gamma'^*$, and $\perp$ is a character in s (this follows from the fact that popping $\perp$ can happen only in a transition from q_{accept} to itself). From the fact that this is the last i.d. not in q_{accept}, we have that q must be in A (by the definition of Δ'). From the fact that there is exactly one transition involving popping of $\perp$ and it is after this i.d., we have the fact that the part of the run from the second i.d. to this i.d. doesn't involve pushing or popping $\perp$ onto/from the stack, and $\perp$ is on the stack (moreover it is at the bottom position).

From here, we know that the original run is of the form

$$(q_{init}, x, Z) \vdash_{P'} (q_0, x, \perp) \vdash^*_{P'} (q, \epsilon, s) \vdash^*_{P'} (q_{accept}, \epsilon, \epsilon)$$

where the part $(q_0, x, \perp) \vdash^*_{P'} (q, \epsilon, s)$ always has $\perp$ at the bottom of the stack, and no transition in this part involves $\perp$, and $q \in A$. Moreover, there are no transitions in this part which are in Δ' but not in Δ, since q_{init} is always a source and never a sink and $q_0 \neq q_{init}$, and q_{accept} is only in a suffix of the run that starts after this part of the run gets over.

Z is also absent in this part of the run, since Z is involved in only the first transition in the run and nowhere else.

From this information, it follows that $(q_0, x, \perp) \vdash^*_P (q, \epsilon, s)$ (note that $\perp, Z$ are not in the stack alphabet of P, but we resort to a slight abuse of notation here, since we are guaranteed that all transitions in this part of the run are in Δ, and no transition in Δ involves $\perp, Z$ at all – we shall do this in later parts too when it is obvious), where all transitions in this part are in Δ, and $s = s'\perp$ for some $s \in \Gamma^*$ (note that s doesn't contain Z).

Hence, it follows that $(q_0, x, \epsilon) \vdash^*_P (q, \epsilon, s')$ for $q \in A$, whence we are done. □

2. Consider any empty stack PDA $P' = (Q', \Sigma, \Gamma', \Delta', q'_0, A', Z)$. We shall add an extra symbol $\perp$ to Γ', and add some new states and relevant transitions to Δ', and change q'_0, A' too, to get a PDA $P = (Q, \Sigma, \Gamma, \Delta, q_{init}, \{q_{accept}\})$ as follows:

1. $Q = Q' \uplus \{q_{init}, q_{start}, q_{accept}\}$.
2. $\Gamma = \Gamma' \uplus \{\perp\}$.
3. $\Delta = \Delta' \cup \{(q_{init}, \epsilon, \epsilon, q_{start}, \perp), (q_{start}, \epsilon, \epsilon, q_0, Z)\} \cup \{(q, \epsilon, \perp, q_{accept}, \epsilon) \mid \forall q \in Q'\}$.

We prove the claim in the problem in two parts.

Claim 2.3

If P accepts a string x, then P' accepts x.

Proof. Consider the "accepting run" of P on x. We have $(q_{init}, x, \epsilon) \vdash_P (q_{start}, x, \perp) \vdash_P (q_0, x, Z\perp) \vdash^*_P (q, \epsilon, \perp) \vdash_P (q_{accept}, \epsilon, \epsilon)$.

We first show why the first, second and the last transitions are as claimed, as well as why the last two i.d.s are as claimed, below:

Note that the first two transitions are always as specified since there is no path from q_{init} to q_{accept} that doesn't go through q_0 as well as q_{start}.

Note that q_{accept} can only be a sink state in a transition, so q_{accept} can never be in the middle part of the above run, and hence $\perp$ is never popped off the stack in the middle part of the run. So the part of the stack below $\perp$ always remains the same throughout the run (i.e., it always remains empty), so when the final state is reached, we should have run out of the string, and since we reach q_{accept}, $\perp$ must have been popped out of the stack, and this implies (from our previous sentence) that the stack must be empty at the last state. This shows why the run is as claimed.

Now note that from the same argument, q_{init} and q_{accept} are never present in any sequence of i.d.s that gives rise to $(q_0, x, Z\perp) \vdash^*_P (q, \epsilon, \perp)$ (since these states can only act as a source and a sink respectively), and q_{start} is never in any sequence of i.d.s that gives rise to the same relation because of never being a sink of any transition other than the one in Δ' but not in Δ, and hence in such a sequence of i.d.s, there is no transition between consecutive i.d.s that is in Δ' but not in Δ. These two observations, combined with the fact that Δ has no transitions which concern $\perp$, imply that $(q_0, x, Z\perp) \vdash^*_{P'} (q, \epsilon, \perp)$ without

ever popping $\perp$ off the stack, and hence using a simple induction, we have $(q_0, x, Z) \vdash_{P'}^* (q, \epsilon, \epsilon)$ for some $q \in Q'$, which means that P' accepts x, as required. $\square$

Claim 2.4

If P' accepts a string x, then P accepts x.

Proof. The proof is very similar to the previous part; here, we instead consider an "accepting run" of P' on x, say $(q_0, x, Z) \vdash_{P'}^* (q, \epsilon, \epsilon)$ for some $q \in Q'$. Note that since $\Delta' \subseteq \Delta$, we have $(q_0, x, Z) \vdash_P^* (q, \epsilon, \epsilon)$ as well. By a simple induction as mentioned in the previous part, we have that this implies that $(q_0, x, Z\perp) \vdash_P^* (q, \epsilon, \perp)$. Noting that $(q_{init}, x, \epsilon) \vdash_P (q_{start}, x, \perp) \vdash_P (q_0, x, Z\perp)$ and $(q, \epsilon, \perp) \vdash_P (q_{accept}, \epsilon, \epsilon)$, we see that $(q_{init}, x, \epsilon) \vdash_P^* (q_{accept}, \epsilon, \epsilon)$, from where it follows that P accepts x. $\square$

Problem 3

Prove that the class of context-free languages is closed under intersection with regular languages. That is, prove that if L_1 is a context-free language and L_2 is a regular language, then $L_1 \cap L_2$ is a context-free language.

Solution.
Let the PDA associated with L_1 be $P = (Q_1, \Sigma, \Gamma, q_{01}, \Delta_1, A_1)$ and the DFA associated with L_2 be $D = (Q_2, \Sigma, \delta_2, q_{02}, A_2)$.

Now consider a PDA $P' = (Q, \Sigma, \Gamma, q_0, \Delta, A)$, where

1. $Q = Q_1 \times Q_2$
2. $q_0 = (q_{01}, q_{02})$
3. $\Delta_{01}= \{((q_1, q_2), a, B, (q_3, q_4), B') \mid \delta(q_2, a) = q_4, (q_1, a, B, q_3, B') \in \Delta_2 \}$
4. $\Delta_{02}= \{((q_1, q_2), \epsilon, B, (q_3, q_2), B') \mid (q_1, \epsilon, B, q_3, B') \in \Delta_1 \}$
5. $\Delta = \Delta_{01} \cup \Delta_{02}$
6. $A = \{(q_1, q_2) \mid q_1 \in A_1, q_2 \in A_2 \}$

Claim 3.1

$L(P') = L_1 \cap L_2$

Proof.

Claim 3.2

$L_1 \cap L_2 \subseteq L(P')$

Proof. Let $x \in L_1 \cap L_2$, now as x is in L_1 and L_2, therefore there exist accepting runs of both P and D on x.

Now we perform induction on the length of string (say n).

Induction Hypothesis $P(n)$: for any $x \in \Sigma^*$ with $|x| = n$, if $\hat{\delta}(q_{02}, x) = q_2'$ and $(q_{01}, x, \epsilon) \rightarrow_P (q_1', \epsilon, \alpha)$ then $((q_{01}, q_{02}), x, \epsilon) \rightarrow_{P'} ((q_1', q_2'), \epsilon, \alpha)$

Base Case $n = 0$, therefore $x = \epsilon$, hence $q_2' = q_{02}$, now as $(q_{01}, \epsilon, \epsilon) \rightarrow_P (q_1', \epsilon, \alpha)$ therefore all transitions take ϵ as input. Now considering transitions in Δ_{02} and following the same transitions as followed in P, (the second element in the state tuple remains the same throughout the run) and thus we can reach the the ID $((q_1', q_2'), \epsilon, \alpha)$ and hence $(q_{01}, q_{02}), x, \epsilon) \rightarrow_{P'} ((q_1', q_2'), \epsilon, \alpha)$, as needed.

Inductive Case For $n > 0$, let $x_1 = x[1, 2, ..n-1]$. There exist q_3, q_4 such that $\hat{\delta}(q_{02}, x_1) = q_4$ and $\delta(q_4, x[n]) = q_2'$ and $(q_{01}, x_1, \epsilon) \rightarrow_P^* (q_3, \epsilon, \alpha')$ and $(q_3, x[n], \alpha') \rightarrow_P^* (q_1', \epsilon, \alpha)$. Now as x_1 is of length $n-1$, and $P(n-1)$ is true, so $((q_{01}, q_{02}), x_1, \epsilon) \rightarrow_{P'} ((q_3, q_4), \epsilon, \alpha')$.

Now during $(q_3, x[n], \alpha') \rightarrow_P^* (q_1', \epsilon, \alpha)$ one transition takes $x[n]$ as input and all the other transition take ϵ as the input. Now take all the transition till the transition corresponding to $x[n]$. Corresponding to these transitions we get transitions in Δ_{02} where the only first state of state tuple changes. Now take the transition where $x[n]$ is the input. Corresponding to this there is a transition in Δ_{01}. Then again take the ϵ transitions as done previously.

So $((q_3, q_4), \epsilon, \alpha') \rightarrow_P^* ((q_1', q_2'), \epsilon, \alpha')$, Combining this with $(q_{01}, q_{02}), x_1, \epsilon) \rightarrow_{P'}^* ((q_3, q_4), \epsilon, \alpha')$ we get $(q_{01}, q_{02}), x_1, \epsilon) \rightarrow_{P'}^* ((q_1', q_2'), \epsilon, \alpha')$.

Now as $x \in L_1 \cap L_2$, therefore $\hat{\delta}(q_{02}, x) = q_2'$ where $q_2' \in A_2$ and there exists a run $(q_{01}, x, \epsilon) \rightarrow_P^* (q_1', \epsilon, \alpha)$ where $q_1' \in A_1$. Now using the above inductive claim $(q_{01}, q_{02}), x, \epsilon) \rightarrow_{P'}^* ((q_1', q_2'), \epsilon, \alpha)$, but $(q_1', q_2') \in A$ as $q_1 \in A_1, q_2 \in A_2$. Therefore x is accepted by P', and hence $x \in L(P')$ and therefore $L_1 \cap L_2 \subseteq L(P')$

□

Claim 3.3

$L(P') \subseteq L_1 \cap L_2$

Proof. Consider a string $x \in L(P')$. Consider an accepting run of P' on x. We perform induction on number of transitions.

Induction Hypothesis $P(n)$: if $((q_1, q_2), x, \alpha) \rightarrow^*_{P'} ((q_1', q_2'), \epsilon, \alpha')$ in at most n transitions, then $\hat{\delta}(q_2, x) = q_2'$ and $(q_1, x, \alpha) \rightarrow^*_P (q_1', \epsilon, \alpha')$

Base Case $n = 0$ – here $x = \epsilon, q_1' = q_1, q_2' = q_2, \alpha = \alpha'$. This is trivial as $\hat{\delta}(q_2, x) = q_2$ because of $q_2' = q_2$, and in P, $(q_1, x, \alpha) \rightarrow^*_P (q_1, \epsilon, \alpha')$

Inductive Case For $n > 0$, there exist x', q_3, q_4 and α' such that $x = ax'$ for some $a \in \Sigma \cup \{\varepsilon\}, x' \in \Sigma^*$

$$((q_1, q_2), x, \alpha) \vdash_P ((q_3, q_4), x', \alpha'') \rightarrow^*_P ((q_1', q_2'), \epsilon, \alpha')$$

The first transition can be of two type according to in which subdelta it is present

1. First transition belong to Δ_{01}, so $\delta(q_2, a) = q_4$, $(q_1, a, B, q_3, B') \in \Delta_1$, where B and B' are such that after poping B and pushing B' to α we get α''
   now as P(n-1) holds so $\hat{\delta}(q_4, x') = q_2'$ and now using $\delta(q_2, a) = q_4$ we get $\hat{\delta}(q_2, x) = q_2'$. Now as P(n-1) holds therefore $(q_3, x', \alpha'') \rightarrow^*_P (q_1', x', \alpha')$, now the transition $(q_1, a, \alpha) \vdash (q_3, \epsilon, \alpha'')$ this is possible as $(q_1, a, B, q_3, B') \in \Delta_2$ combining both we get $(q_1, x, \alpha) \rightarrow^*_P (q_1', \epsilon, \alpha')$ hence induction holds in this case

2. Transition belongs to Δ_{02}, so $a = \epsilon$, $q_2 = q_4$, $(q_1, a, B, q_3, B') \in \Delta_1$, where B and B' are such that after popping B and pushing B' to α we get α''
   Now as P(n-1) holds $\hat{\delta}(q_4, x') = q_2'$, now as $q_4 = q_2$ and $x = x'$, therefore $\hat{\delta}(q_2, x_1) = q_2'$.
   Now as P(n-1) holds therefore $(q_3, x', \alpha'') \rightarrow^*_P (q_1', x', \alpha')$, now the transition $(q_1, a, \alpha) \vdash (q_3, \epsilon, \alpha'')$ this is possible as $(q_1, a, B, q_3, B') \in \Delta_2$ combining both we get $(q_1, x, \alpha) \rightarrow^*_P (q_1', \epsilon, \alpha')$ hence induction holds in this case

As all the above cases are exhaustive therefore the induction holds.
Now as $x \in L(P')$, therefore $((q_{01}, q_{02}), x, \epsilon) - >^*_{P'} ((q_1', q_2'), \epsilon, \alpha)$ where $q_1' \in A_1, q_2' \in A_2$ Now from the above claim $\hat{\delta}(q_{02}, x) = q_1'$ hence accepting x, i.e. $x \in L_2$, also from the above claim $(q_{01}, x, \epsilon) - >^*_P (q_1', \epsilon, \alpha)$ and hence $x \in L_2$, combining both we get $x \in L_1 \cup L_2$ which further implies $L(P') \subseteq L_1 \cap L_2$

□

Combining both of the above claims be get $L(P') = L_1 \cap L_2$

□

Problem 4

Consider the following two languages over the alphabet $\{a, b, c, d\}$.

$$L_1 = \{a^m b^n c^m d^n \mid m, n \in \mathbb{N} \cup \{0\}\},$$

$$L_2 = \{a^m b^n c^n d^m \mid m, n \in \mathbb{N} \cup \{0\}\}.$$

Determine whether each of these languages is context-free, and prove your answer.

Solution.

Claim 4.1

L_1 is not context-free

Proof. Suppose L_1 is context-free. Then it must satisfy the pumping lemma for context-free languages. But as we will show in the claim below L_1 does not satisfy the pumping lemma. Hence L_1 can not be context-free.

Claim 4.2

For all $p \in \mathbb{N}$ there exists a string $w \in L_1$ such that $|w| \geq p$ and for all strings $u, v, x, y, z \in \Sigma^*$ such that $w = uvxyz$ and one of the following does not hold

1. $|vy| > 0$
2. $|vxy| \leq p$
3. $uv^i xy^i z \in L_1$ for all $i \in \mathbb{N} \cup \{0\}$.

Proof. For any $p \in \mathbb{N}$ consider the string $w = a^p b^p c^p d^p$ and its partition $w = uvxyz$. Then we have 3 cases

1. vxy is a substring of $a^p b^p$. In this case if we assume property 1 and 2 then by pumping once (i.e. $i = 2$) we have that $uv^2xy^2z \notin L_1$ because number of a's increase while number of c's remain the same. Hence property 3 is violated here.
2. vxy is a substring of $b^p c^p$. In this case if we assume property 1 and 2 then by pumping once (i.e. $i = 2$) we have that $uv^2xy^2z \notin L_1$ because number of b's increase while number of d's remain the same. Hence property 3 is again violated here.
3. vxy is a substring of $c^p d^p$. In this case if we assume property 1 and 2 then by pumping once (i.e. $i = 2$) we have that $uv^2xy^2z \notin L_1$ because number of c's increase while number of a's remain the same. Hence property 3 is again violated here.

Thus for each case we have proved the claim. □

Hence L_1 can not be context-free. □

Claim 4.3

L_2 is context-free

Proof. Consider the language of grammar of $G = (N, \Sigma, R, S)$ where $N = \{S, S'\}$, and $\Sigma = \{a, b, c, d\}$

and following rules under R:

$$S \to \varepsilon$$
$$S \to aSd$$
$$S \to S'$$
$$S' \to \varepsilon$$
$$S' \to bS'c$$

Claim 4.4
$L_2 \subseteq \mathcal{L}(G)$

Proof. Consider any string $s \in L_2$ as $a^m b^n c^n d^m$. Then to generate this string via grammar G we may apply the production rule 2 m times, followed by production rule 3 and then production rule 5 applied n times finally followed by production rule 4. In the corner case when $m, n = 0$ we may simply apply production rule 1 and be done with it. □

Claim 4.5
$\mathcal{L}(G) \subseteq L_2$

Proof.

Claim 4.6
If $S' \overset{*}{\Longrightarrow} s$ then $s = b^n c^n$ for some $n \in \mathbb{N} \cup \{0\}$

Proof. Proof by induction on number of production rules used in shortest derivation.

Base Case Number of production rules used = 1, then we have must have used the rule 4 giving us $s = \varepsilon$.

Inductive Case If we have used more than one production rule then our first production rule must have been $S' \implies bS'c \overset{*}{\Longrightarrow} bb^n c^n c$ where the derivation follows from inductive hypothesis. □

Claim 4.7
If $S \overset{*}{\Longrightarrow} s$ then $s = a^m b^n c^n d^m$ for some $m, n \in \mathbb{N} \cup \{0\}$

Proof. Proof by induction on number of production rules used in shortest derivation.

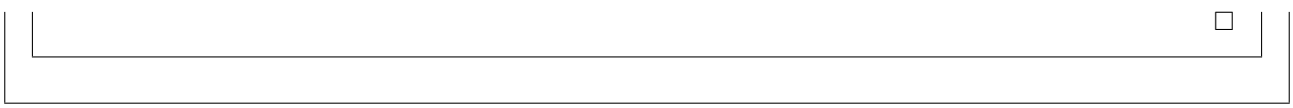
Base Case Number of production rules used = 1, then we have must have used the rule 1 giving us $s = \varepsilon$.

Inductive Case If we have used more than one production rule then our first production rule must have been either

1. $S \implies aSd \overset{*}{\Longrightarrow} aa^m b^n c^n d^m d$ where the derivation follows from inductive hypothesis.
2. $S \implies S' \overset{*}{\Longrightarrow} b^n c^n$ where the derivation follows from the previous claim. Here $m = 0$.

□

□



Problem 5

Prove that the language $\{a^m b^n \mid n \text{ is a multiple of } m\} \subseteq \{a, b\}^*$ is not context-free.

Solution.

Claim 5.1

L_1 is not context-free

Proof. Suppose L is context-free. Then it must satisfy the pumping lemma for context-free languages. But as we will show in the claim below L does not satisfy the pumping lemma. Hence L can not be context-free.

Claim 5.2

For all $p \in \mathbb{N}$ there exists a string $w \in L$ such that $|w| \geq p$ and for all strings $u, v, x, y, z \in \Sigma^*$ such that $w = uvxyz$ and one of the following does not hold

1. $|vy| > 0$
2. $|vxy| \leq p$
3. $uv^i xy^i z \in L$ for all $i \in \mathbb{N} \cup \{0\}$.

Proof. For any $p \in \mathbb{N}$ take p' as p^{th} prime. consider the string $w = a^{2p'} b^{4p'^2}$ and its partition $w = uvxyz$. Now consider the string vy – it can be represented as $a^r b^s$, where $0 \leq r \leq p$, $0 \leq s \leq p$ but both are not 0 at the same time if the property 1 and 2 are to be satisfied.
Then we have 3 cases

1. $r = 0$ – in this case take $i = 2$, then the string $uv^2xy^2z \notin L$ as # of b would be $4p'^2 + s$ and # of a would be remain same that is $2p'$. Now $0 \leq s \leq p < p'$ and also as r is 0, therefore s cannot be 0, so # of b is not divisible by # of a. Hence property 3 is violated here.

$$4p'^2 + s \neq 0 \mod 2p'$$

2. $s = 0$ – in this case take $i = 4p'^2 + 1$, then the string $uv^{4p'^2+1}xy^{4p'^2+1}z \notin L$ as # of a would be $2p' + 4p'^2 \cdot r$ and # of b would remain the same, that is, $4p'^2$. Now $0 \leq r \leq p$ and also as s is 0, therefore r cannot be 0, so # of $a \geq 2p' + 4p'^2$ and hence # of b is not divisible by # of a. Hence property 3 is violated here.

3. $s \neq 0$, $r \neq 0$, In this case take $i = 4p'^2 + 1$, then the string $uv^{4p'^2+1}xy^{4p'^2+1}z \notin L$ as # of a would be $2p' + 4p'^2 \cdot r$ and # of a would be that is $4p'^2 + 4p'^2 \cdot s$.
Now let say # of b is divisible by # of a. i.e.,

$$(4p'^2 + 4p'^2 \cdot s) \equiv 0 \mod (2p' + 4p'^2 \cdot r)$$

Dividing $2p'$ from both sides

$$(2p'(1 + s)) \equiv 0 \mod (1 + 2p' \cdot r)$$

Now note that $(1 + 2p' \cdot r)$ does not have either 2 or p' as a factor therefore for the above equivalence to hold we must have

$$(1 + s) \equiv 0 \mod (1 + 2p' \cdot r)$$

But $(1 + 2p' \cdot r) \geq 1 + 2p' > 1 + p' > 1 + s$ this is a contradiction. Hence our assumption was wrong and # of b is divisible by # of a. Hence property 3 is again violated here.

□

Hence L can not be context-free. $\square$