CS 181 HW2 2021 CS181

CHARLES ZHANG

TOTAL POINTS

20 / 20

QUESTION 1

1 NFA 5 / 5

✓ **- 0 pts Correct**

- 2 pts NFA does not produce the correct language

- 2 pts Nondeterminism could be used more effectively

- 5 pts No attempt

QUESTION 2

2 Pumping Lemma 0 / 0

✓ **+ 0 pts Postponed to next week - not graded**

+ 1 pts Correct & clear

QUESTION 3

3 Proof via Closure Properties 5 / 5

✓ **+ 5 pts Correct**

+ 3 pts Partially Correct

+ 4 pts Almost Correct

+ 1 pts Attempted, but did not try to justify in a proper manner.

+ 0 pts Not Attempted

QUESTION 4

4 Regular Expression 4 / 4

✓ **- 0 pts Correct**

- 1 pts ****Minor mistake, cannot generate some of the following strings. We test your solution using following strings:

- 0
- \#
- 11
- \#\#
- \#\#11
- 11\#\#
- 00
- 0011
- 001100
- 110011
- 1100
- \#\#00
- \#\#0011
- 1101\#\#
- 11011\#\#
- 0011\#\#
- 0011\#\#01\#
- \#\#0011011\#
- \#\#110011011\#
- 01011\#
- 01\#01011\#
- 01\#010000110101\#

- 1 pts Minor mistake, your regular expression will generate some strings that are not in the language. Tested strings prefix contains:

- 0\#
- 00##
- 110##

- 1 pts Do not have any explanation, or only have limited/inadequate explanation.

- 0.5 pts Informal format of regular expression,

- 4 pts Totally wrong, or no regular expression provided.

QUESTION 5

5 Inductive Proof on Tree 6 / 6

+ 0 pts No answer

+ 2 pts Partially correct / Some logic errors / Missing justifications

✓ **+ 4 pts Mostly correct / Minor logic errors or missing justification**

✓ **+ 6 pts Correct and Clear**

I can figure out that "T's subtrees" means the subtrees rooted at each of the root's children, but the reader should not have to figure that out. Please be more careful & complete when describing the elements of your argument.

2 not clear what this means. you should pick a name for whatever you are using as the induction value and be clear about it.

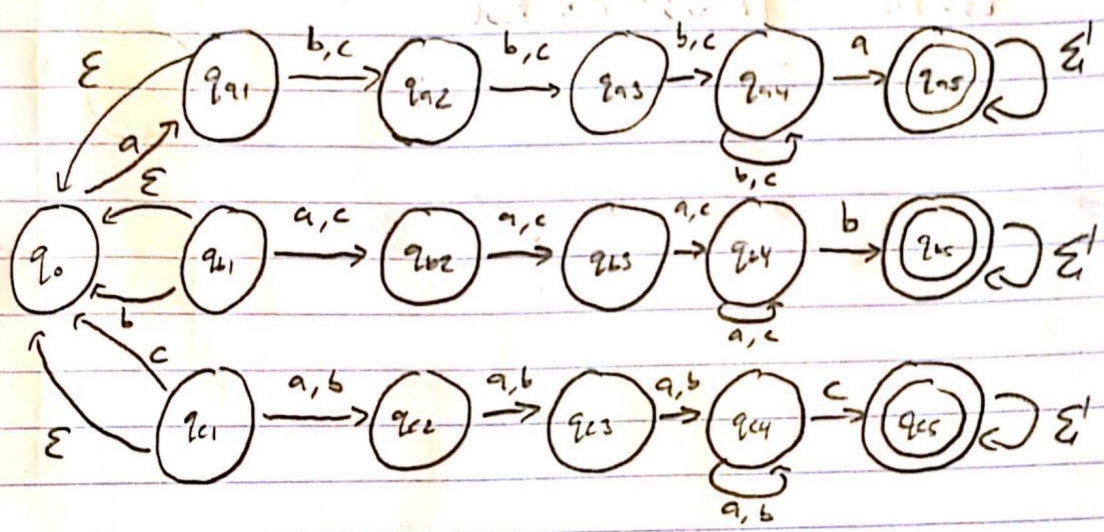
3 I think I know what you mean, but there is some hidden logic here that should be spelled-out.

Charles Zhang
305413659

CS181 HW#2

1) $\Sigma' = \{a, b, c\}$

$L_1 = \{w \in \Sigma'^* \mid w$ contains at least one substring of two of the same symbol separated by at least three occurrances of the other two symbols$\}$



This design splits the language into 3 parts: substrings that begin with a, those that begin with b, and those that begin with c. It uses 3 intermediary states for each case to fulfill the condition that the start/end symbol are separated by 3+ occurrences of the other 2 symbols. At the 3rd state (q_{a4}, q_{b4}, q_{c4}), the design loops into the same state if it receives anything other than the symbol the substring starts with (accounting for 3+) and enters an accepting state if the target symbol is read. The accepting states loop into themselves, since, once a valid substring is found, the string is in the language. This makes use of nondeterminism by using the blocking convention and ε-arrows to simplify the design.

1 NFA 5 / 5

✓ **- 0 pts** **Correct**

- 2 pts NFA does not produce the correct language

- 2 pts Nondeterminism could be used more effectively

- 5 pts No attempt

2) Cancelled

3) $L_3 = \{ w \in \Sigma^* \mid \#(a,w) = 2\#(b,w) \}$

$\Sigma = \{a, b\}$

$L_2 = \{ a^{(2n)} b^n \mid n \geq 0 \}$ → not FSL

Union, ~~Concatenation~~, Kleene

Complementation, ∩ intersection ✓

~~Reverse~~

Proof:

- Assume L_3 is regular
- By closure properties of FSL, L_3 must be closed under intersection
- By the notation of regular expressions, $L_3 \cap a^*b^* = \{ a^{(2n)} b^n \mid n \geq 0 \}$
- In other words $L_3 \cap a^*b^* = L_2$
- a^*b^* is regular by closure under Kleene star and concatenation
- Therefore, if L_3 is regular, $L_3 \cap a^*b^*$ is regular by closure under intersection
- However, $L_3 \cap a^*b^* = L_2$, and L_2 is known to be nonregular, leading to a contradiction →←
- L_3 cannot be regular by contradiction

2 Pumping Lemma 0 / 0

✓ **+ 0 pts** **Postponed to next week - not graded**

+ 1 pts Correct & clear

2) Cancelled

3) $L_3 = \{ w \in \Sigma^* \mid \#(a,w) = 2\#(b,w) \}$

$\Sigma = \{a, b\}$

$L_2 = \{ a^{(2n)} b^n \mid n \geq 0 \}$ → not FSL

Union, ~~Concatenation~~, Kleene

Complementation, ∩ intersection ✓

~~Reverse~~

Proof:

- Assume L_3 is regular
- By closure properties of FSL, L_3 must be closed under intersection
- By the notation of regular expressions, $L_3 \cap a^*b^* = \{ a^{(2n)} b^n \mid n \geq 0 \}$
- In other words $L_3 \cap a^*b^* = L_2$
- a^*b^* is regular by closure under Kleene star and concatenation
- Therefore, if L_3 is regular, $L_3 \cap a^*b^*$ is regular by closure under intersection
- However, $L_3 \cap a^*b^* = L_2$, and L_2 is known to be nonregular, leading to a contradiction →←
- L_3 cannot be regular by contradiction

3 Proof via Closure Properties 5 / 5

✓ **+ 5 pts Correct**

+ 3 pts Partially Correct

+ 4 pts Almost Correct

+ 1 pts Attempted, but did not try to justify in a proper manner.

+ 0 pts Not Attempted

4) $\Sigma' = \{0, 1, \#\}$

$L_4 = \{w \in \Sigma'^* \mid$ in w, to the right of any 0s, there is at least one 1 before any #'s$\}$

~~$(1 \cup \#)^* (0 (0 \cup 1)^* (1 \# (1 \cup \#)^*)^*)^*$~~

$(1 \cup \#)^* (0 (0^* (1 \# (1 \cup \#)^* \cup 1)^*)^*)^*$

This reg. exp. first checks for any prefixes of w that don't contain a 0, which should all be accepted. If it finds a 0, it then accepts any 0s that follow it. Afterwards, it looks for a 1# or a 1. If a # is found instead, the string will not be accepted. If a 1# is found, it will then accept everything past it until it hits another 0, where it once again must check the language's condition. If a 1 is found, it will loole for a 1# or 1 again.

5) If T has $k^h + 1$ leaf nodes, then T has height of $h+1$

Let T be a k-ary directed rooted tree. Show by induction on h that for any degree $k \geq 1$

Basis: $h = 2$

- T has $k^2 + 1$ leaf nodes at a height of 3.
- Since each node has at most k children, the most leaf nodes possible at height 2 is $k \cdot k$, or k^2.
- Since the number of leaf nodes is $k^2 + 1$, the minimum height possible is 3.
- Basis solved ✓

4 Regular Expression 4 / 4

✓ **- 0 pts Correct**

- 1 pts ****Minor mistake, cannot generate some of the following strings. We test your solution using following strings:

- 0
- \#
- 11
- \#\#
- \#\#11
- 11\#\#
- 00
- 0011
- 001100
- 110011
- 1100
- \#\#00
- \#\#0011
- 1101\#\#
- 11011\#\#
- 0011\#\#
- 0011\#\#01\#
- \#\#0011011\#
- \#\#110011011\#
- 01011\#
- 01\#01011\#
- 01\#010000110101\#

- 1 pts Minor mistake, your regular expression will generate some strings that are not in the language. Tested strings prefix contains:

- 0\#
- 00##
- 110##

- 1 pts Do not have any explanation, or only have limited/inadequate explanation.

- 0.5 pts Informal format of regular expression,

- 4 pts Totally wrong, or no regular expression provided.

4) $\Sigma' = \{0, 1, \#\}$

$L_4 = \{w \in \Sigma'^* \mid$ in w, to the right of any 0s, there is at least one 1 before any #'s$\}$

~~$(1 \cup \#)^* (0 (0 \cup 1)^* (1 \# (1 \cup \#)^*)^*)^*$~~

$(1 \cup \#)^* (0 (0^* (1 \# (1 \cup \#)^* \cup 1)^*)^*)^*$

This reg. exp. first checks for any prefixes of w that don't contain a 0, which should all be accepted. If it finds a 0, it then accepts any 0s that follow it. Afterwards, it looks for a 1# or a 1. If a # is found instead, the string will not be accepted. If a 1# is found, it will then accept everything past it until it hits another 0, where it once again must check the language's condition. If a 1 is found, it will loole for a 1# or 1 again.

5) If T has $k^h + 1$ leaf nodes, then T has height of $h+1$

Let T be a k-ary directed rooted tree. Show by induction on h that for any degree $k \geq 1$

Basis: $h = 2$

- T has $k^2 + 1$ leaf nodes at a height of 3.
- Since each node has at most k children, the most leaf nodes possible at height 2 is $k \cdot k$, or k^2.
- Since the number of leaf nodes is $k^2 + 1$, the minimum height possible is 3.
- Basis solved ✓

Induction hypothesis: Assume that for a tree with k^x+1 leaf nodes, where $x \geq 1$, the tree must have a height of at least $x+1$

Inductive step: Prove that for a tree with $k^{x+1}+1$ leaf nodes, the minimum height of that tree is $x+2$

- Since height is h+1 in all cases, an induction on the height acts as an induction on h
- By the induction hypothesis, we know a tree of height x can have at most k^x leaf nodes, since adding any more leaf nodes would increase the height of the tree to x+1
- Let T' be a k-ary tree of height x+1
- By the definition of a tree, T's subtrees have a height $\leq x$
- Using the induction hypothesis, we know these subtrees have at most k^x leaf nodes each
- By the definition of a k-ary tree, at most k such subtrees exist in T'
- By the nature of trees, the total number of leaf nodes in T' is equal to the number of leaf nodes in its subtrees
- Having at most k subtrees with k^x leaf nodes in each subtree tells us that T' has at most $k(k^x)$, or k^{x+1} leaf nodes
- Therefore, a tree with a height of x+1 can have at most k^{x+1} leaf nodes
- Therefore, for a tree to have $k^{x+1}+1$ leaf nodes, it must have a height of x+2
- The property holds for trees of height x+2
- Induction solved ✓

5 Inductive Proof on Tree 6 / 6

+ **0 pts** No answer

+ **2 pts** Partially correct / Some logic errors / Missing justifications

✓ **+ 4 pts** **Mostly correct / Minor logic errors or missing justification**

✓ **+ 6 pts** **Correct and Clear**

1 I can figure out that "T's subtrees" means the subtrees rooted at each of the root's children, but the reader should not have to figure that out. Please be more careful & complete when describing the elements of your argument.

2 not clear what this means. you should pick a name for whatever you are using as the induction value and be clear about it.

3 I think I know what you mean, but there is some hidden logic here that should be spelled-out.