CSC 456/656 Spring 2023 Answers to Second Examination March 8

1. True or False. T = true, F = false, and O = open, meaning that the answer is not known science at this time.

   (i) **T** Every regular language is in Nick's class.

   (ii) **T** Every context-free language is in Nick's class.

   (iii) **T** If L_1 is $\mathcal{NP}$ and L_2 is $\mathcal{NP}$–complete, there must be a $\mathcal{P}$–TIME reduction of L_1 to L_2.

   The definition of $\mathcal{NP}$–completeness.

   (iv) **T** The set of binary numerals for prime numbers is a polynomial time language.

   That was only recently proved. Before that, the correct answer would have been **O**.

   (v) **T** The complement of any $\mathcal{P}$-TIME language is $\mathcal{P}$-TIME.

   Any deterministic "–TIME" class is closed under complementation.

   (vi) **O** The complement of any $\mathcal{NP}$ language is $\mathcal{NP}$.

   This is true if $\mathcal{P} = \mathcal{NP}$.

   (vii) **T** The complement of any $\mathcal{P}$–SPACE language is $\mathcal{P}$–SPACE.

   Any "–SPACE" class is closed under complementation.

   (viii) **T** The complement of any recursive (that is, decidable) language is recursive.

   (ix) **T** The complement of any undecidable language is undecidable.

   That actually follows from the previous answer.

   (x) **F** If L is a language and L^* is a regular language, then L must be a regular language.

   If L is any language over an alphabet Σ such that $\Sigma \subseteq L$, then $L^* = \Sigma^*$, which is regular.

   (xi) **T** For any infinite countable sets A and B, there is a 1-1 correspondence between A and B.

   There is only one countable infinity.

   (xii) **F** For any uncountable sets A and B, there is a 1-1 correspondence between A and B.

   There are infinitely many uncountable infinities. If S is any set, 2^S is larger. For example, there are more subsets of real numbers than there are real numbers.

   (xiii) **T** A language L is recursively enumerable if and only if there is a machine which accepts L.

   I have given the proof of that in class, and will give it again.

   (xiv) **F** Given any real number x, let L_x be the set of all fractions whose values are less than x. Then there must be a machine that decides L_x.

If there is a machine that decides L_x, then the decimal expansion of x is computable, hence x is a recursive real number. But there are uncountably many real numbers, and only countably many recursive real numbers.

(xv) **O** $\mathcal{P}$–TIME $= \mathcal{NP}$.

The classic $\mathcal{P} = \mathcal{NP}$ problem.

(xvi) **O** $\mathcal{P}$–TIME $= \mathcal{NC}$.

Recently, it has become clear, with the increased importance of multi-processor computers, that the $\mathcal{P} = \mathcal{NC}$ problem is also of fundamental importance. "All experts" believe that they are not equal, but no proof is known.

(xvii) **T** Every $\mathcal{NP}$ language is reducible to SAT in polynomial time.

This follows from the answer to (iii) and the fact that SAT is $\mathcal{NP}$–complete.

(xviii) **T** If a Boolean expression is satisfiable, there is a polynomial time proof that it is satisfiable.

If you know a satisfying assignment, you can verify it in linear time.

(xix) **F** A Boolean expression is a tautology if and only if it is not a contradiction.

An expresssion which is not a contradiction is satisfiable, but not necessarily a tautology.

(xx) **T** A language L is decidable if and only if there is some machine which enumerates L in canonical order.

2. [20 points] Let G be the CF grammar given below, where S is the start symbol. Show that G is ambiguous by giving two different rightmost derivations for the string *iiaea*.

1. $S \to iS$
2. $S \to iSeS$
3. $S \to a$

$$S \Rightarrow iS \Rightarrow iiSeS \Rightarrow iiSea \Rightarrow iiaea$$

$$S \Rightarrow iSeSS \Rightarrow iSea \Rightarrow iiSea \Rightarrow iiaea$$

3. [20 points] Walk through the computation of the LALR parser given below, for the grammar given below, where the input string is $(x + y) * x$.

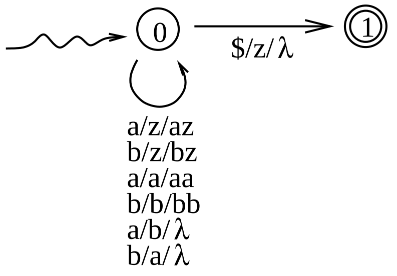
TOP

1. $E \to E +_2 E_3$
2. $E \to E *_4 E_5$
3. $E \to (_6 E_7)_8$
4. $E \to x_9$
5. $E \to y_{10}$

	x	y	$+$	$*$	(	)	\$	E
0	$s9$	$s10$			$s6$			1
1			$s2$	$s4$			HALT	
2	$s9$	$s10$			$s6$			3
3			$r1$	$s4$		$r1$	$r1$	
4	$s9$	$s10$			$s6$			5
5			$r2$	$r2$		$r2$	$r2$	
6	$s9$	$s10$			$s6$			7
7			$s2$	$s4$		$s8$		
8			$r3$	$r3$		$r3$	$r3$	
9			$r4$	$r4$		$r4$	$r4$	
10			$r5$	$r5$		$r5$	$r5$	

STACK	INPUT	OUTPUT	ACTION
$\$_0$	$(x + y) * x\$$		
$\$_0(_6$	$x + y) * x\$$		$s6$
$\$_0(_6x_9$	$+y) * x\$$		$s9$
$\$_0(_6E_7$	$+y) * x\$$	4	$r4$
$\$_0(_6E_7+_2$	$y) * x\$$	4	$s2$
$\$_0(_6E_7 +_2 y_{10}$	$) * x\$$	4	$s10$
$\$_0(_6E_7 +_2 E_3$	$) * x\$$	45	$r5$
$\$_0(_6E_7$	$) * x\$$	451	$r1$
$\$_0(_6E_7)_8$	$*x\$$	451	$s8$
$\$_0E_1$	$*x\$$	4513	$r3$
$\$_0E_1*_4$	$x\$$	4513	$s4$
$\$_0E_1 *_4 x_9$	\$	4513	$s9$
$\$_0E_1 *_4 E_5$	\$	45134	$r4$
$\$_0E_1$	\$	451342	$r2$
		451342	HALT

4. [20 points] Design a DPDA which accepts the language $L = \{w \in \{a, b\}^* : \#_a(w) = \#_b(w)\}$, that is, all strings over $\{a, b\}$ which have equal numbers of a's and b's. An input string must have an end-of-file symbol; for example, *abbaba\$*.



Imagine yourself writing a program to check whether $w \in L$. You could count the a's and b's to see if they are equal. But you can't do that with one stack. The key is that the actual number of each symbol is irrelevant: you need only verify that they agree, which means you need to keep track of the *difference*, $\#_a - \#_b$. Each time you read a, that difference increases, and time you read b it decreases.

The stack holds that many a's. But you can't have negatively many items on the stack, so you store b's to represent negative a's. If you read a, it will cancel a b popped off the stack, and the other way around. If you read a and pop a, you have to push the a but save the other a; likewise b. The string is accepted if the stack is empty and there is no more input If you read a, it will cancel a b popped off the stack, and they other way around. If you read a and pop a, you have to push the a but save the other a; likewise b. The string is accepted if the stack is empty and there is no more input.

5. [20 points] Prove that every decidable language can be enumerated in canonical order by some machine.

Let L be a decidable language over an alphabet Σ. Let $w_1, w_2, \ldots$ be the canonical order enumeration of Σ^*. The following program enumerates L in canonical order:

```
For all positive integers $i$
    If ($w_i \in L$) write $w_i$
```

6. [20 points] State the pumping lemma for context-free languages *correctly.* Pay close attention to your logic, including the order in which you write the quantifiers. If you have all the correct words in the wrong order, but your logic is wrong, you might get no credit.

For any context-free language L
There exists an integer p such that
For any $w \in L$ of length at least p
There exist strings u, v, x, y, z such that the following four conditions hold:

1. $w = uvxyz$
2. $|vxy| \leq p$
3. $|v| + |y| \geq 1$
4. For any integer $i \geq 0$, $uv^ixy^iz \in L$

7. [20 points] Prove that the halting problem is undecidable.

Proof by contradiction. Recall that L_{HALT} is the set of all strings $\langle M \rangle w$ such that M is a machine which halts given input w.

Assume the halting problem is decidable. Let $L_{\text{DIAG}} = \{\langle M \rangle : \langle M \rangle \langle M \rangle \notin L_{\text{HALT}}\}$, the *diagonal* language. Let M_{DIAG} be a machine which executes the following algorithm:

```
Read $\langle M \rangle$
 If ($\langle M \rangle \langle M \rangle \in L_{\text{HALT}}$)
    Run forever.
 Else
    HALT.
```

Then M_{DIAG} accepts L_{DIAG} (1)

We now illustrate the contradiction. Either $\langle M_{\text{DIAG}} \rangle \in L_{\text{HALT}}$, or not.
1. If $\langle M_{\text{DIAG}} \rangle \in L_{\text{HALT}}$, then, by (1), the program accepts $\langle M_{\text{DIAG}} \rangle$.
2. If $\langle M_{\text{DIAG}} \rangle \notin L_{\text{HALT}}$, then the program will run forever with input $\langle M_{\text{DIAG}} \rangle$, because the **If** condition is satisfied: thus does not accept $\langle M_{\text{DIAG}} \rangle$.

Contradiction. We conclude that the halting problem is not decidable.