University of Nevada, Las Vegas Computer Science 456/656 Spring 2021

Assignment 6: Due Saturday November 12 2022, 11:59 PM

This is the final version of the assignment.

Name:_____

You are permitted to work in groups, get help from others, read books, and use the internet. Turn in the assignment in the manner given to you by our grader, Janeen Sudiacal.

1. True, false, or open:

   (a) **T** If L_1 and L_2 are languages and there is a $\mathcal{P}$–TIME reduction of L_1 to L_2, and if there is a machine that accepts L_2 in polynomial time, then there must be a machine that accepts L_1 in polynomial time.

   (b) **F** If L_1 and L_2 are languages and there is a $\mathcal{P}$–TIME reduction of L_1 to L_2, and if there is a machine that accepts L_1 in polynomial time, then there must be a machine that accepts L_2 in polynomial time.

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

   (d) **T** If L_1 and L_2 are languages and there is a recursive reduction of L_1 to L_2, and if L_1 is undecidable, then L_2 is undecidable.

   (e) **T** Context-free grammar equivalence is co-RE.

   (f) **T** The set of binary numerals for prime numbers is $\mathcal{P}$–TIME. (This was only recently proven: it was an outstanding problem for over 2000 years.)

   (g) **O** The factoring problem for binary numerals is $\mathcal{P}$–TIME.

   (h) **T** The class of regular languages is closed under intersection.

   (i) **T** co-$\mathcal{P}$–TIME = $\mathcal{P}$–TIME.

   (j) **T** The class of context-free languages is closed under union.

   (k) **F** The class of context-free languages is closed under intersection.

   (l) **T** The complement of any undecidable language must be undecidable.

   (m) **F** Every context-free language can be parsed by an LALR parser.

   (n) **F** If a function $f : \mathcal{N} \to \mathcal{N}$, where $\mathcal{N}$ is the set of natural numbers, has a mathematical definition, then f must be recursive.

   (o) **F** If Σ is any alphabet, the set of all languages over Σ is countable.

   (p) **O** $\mathcal{P} = \mathcal{NP}$.

   (q) **F** 2SAT is known to be $\mathcal{NP}$-complete.

(r) **T** Every language has a canonical order enumeration. (But it might be uncomputable.)

(s) **T** $\sqrt{2}$ is a recursive real number.

(t) **O** There is a $\mathcal{P}$–TIME algorithm which determines whether a given weighted directed graph has a Hamiltonian cycle whose total weight is no greater than a given number. (Given a directed graph G, a Hamiltonian cycle of G is a directed cycle in G that includes every vertex of G exactly once.) This is the Traveling Salesman Problem, which is $\mathcal{NP}$-complete.

(u) **F** It is known that 2-SAT is $\mathcal{NP}$-complete. (2-SAT is $\mathcal{P}$–TIME.)

(v) **T** Every context-free language is in Nick's Class.

(w) **T** Context-free grammar equivalence is co-RE.

(x) **F** Every context-free language is accepted by some LALR parser.

(y) **O** The Circuit Value Problem (CVP) is $\mathcal{NC}$.

(z) **O** If L is any $\mathcal{P}$–TIME language, there is an $\mathcal{NC}$ reduction of CVP to L. (But there is an $\mathcal{NC}$ reduction of L to CVP.)

2. Correctly state (do not prove) the pumping lemma for context-free languages.

   For any context-free language L there is an integer p such that for any $w \in L$ such that $|w| \geq p$ there exist strings u, v, x, y, z such that:
   1. $w = uvxyz$
   2. $|vxy| \leq p$
   3. $|v| + |y| \geq 1$
   4. for any integer $i \geq 0$ $uv^ixy^iz \in L$.

3. Describe a Nick's Class algorithm which finds the maximum of n integers in $O(\log n)$ time, using $n/\log n$ processors.

   You already know an algorithm, which I call *tournament*, which finds the maximum of n items in $O(\log n)$ time using n processors.

   1. Separate the n items into batches of size $\log n$. There will be $n/\log n$) batches.

   2. For each batch, use one processor to find the maximum of that batch. That takes $O(\log n)$ time using $O(n/\log n)$ processors.

   3. You now have $n/\log n$ candidates. Use tournament to find the maximum of those in $O(\log(n/\log n)) = O(\log n)$ time with $n/\log n$ processors.

   That will be the maximum of the original n integers.

   The following statement is false: "We have proved, in class (or we will have by the end of class on November 7) that a dynamic programming problem is $\mathcal{NC}$ if it has logarithmic reachback and every subprogram is $\mathcal{P}$–TIME." Instead, Theorem 1 in the handout *NC.pdf* is true.

   Note that the circuit value problem does not have logarithmic reachback, since an arc can stetch from a starting gate all the way to the final gate. You can use this fact to work problems 4 and 5 below. (Read the handout NC.pdf.)

   Use Theorem 1 in the handout.

4. Prove that addition of binary numerals is $\mathcal{NC}$. (Hint: This is important for computer architecture.)

   Let x and y be the addends. Assume that each has n binary digits, that is, $x = \sum_{i=0}^{n-1} x_i 2^i$ and $y = \sum_{i=0}^{n-1} y_i 2^i$, where x_i and y_i are each either 0 or 1. $x + y = z = \sum_{i=0}^{n} z_i 2^i$. The binary digits z_i are computed as follows.

   (a) $[S_0]$ Let $c_0 = 0$.

   (b) $[S_i]$ For each $0 \leq i < n$, let $z_i = (x_i + y_i + c_i)\%2$, and let $c_{i+1} = (x_i + y_i + c_i)/2$, using truncated integer division.

   (c) $[S_n]$ Let $z_n = c_n$.

   Each processor S_i has reachback at most 1, executes in $O(1)$ time, and outputs 1 bit. Hence by Theorem 1 in the handout *NC.pdf*, the dynamic program is equivalent to an $\mathcal{NC}$ computation.

5. Prove that every regular language is $\mathcal{NC}$.

   This is Theorem 2 of the handout.

6. Prove that every decidable language can be enumerated in canonical order by some machine.

   Let $L \subseteq \Sigma^*$ be a decidable language over an alphabet Σ. Let $w_1, w_2, \ldots$ be a canonical order enumerate of Σ^*.

   The following program enumerates L in canonical order.

   For all i from 1 to ∞
   If($w_i \in L$) Write w_i

   The loop will not get stuck at the If step because L is decidable.

7. (a) State the Church-Turing thesis.

   Any computation by any machine can be done by some Turing Machine.

   (b) Why is the Church-Turing thesis important?

   If there is a proof that no Turing Machine can work a certain problem, then no machine can work that problem. Turing machines are simple, making proofs easier.

8. The CF grammar given below generates $L = \{a^n b^m : n, m \geq 0\}$. S is the start symbol, and there are two other variables, A and B. An LALR parser for that grammar is also given. Walk through the computation of the parser for the input string *aab*.

1. $S \to A_2 B_4$
2. $A \to A_2 a_3$
3. $A \to \lambda$
4. $B \to B_4 b_5$
5. $B \to \lambda$

	ACTION			GOTO		
	a	b	\$	S	A	B
0	$r3$	$r3$	$r3$	1	2	
1			HALT			
2	$s3$	$r5$	$r5$			4
3	$r2$	$r2$	$r2$			
4		$s5$	$r1$			
5		$r4$	$r4$			

$\$_0$	$aab\$$		
$\$_0 A_2$	$aab\$$	$r3$	3
$\$_0 A_2 a_3$	$ab\$$	$s3$	3
$\$_0 A_2$	$ab\$$	$r2$	32
$\$_0 A_2 a_3$	$b\$$	$s3$	32
$\$_0 A_2$	$b\$$	$r2$	322
$\$_0 A_2 B_4$	$b\$$	$r5$	3225
$\$_0 A_2 B_4 b_5$	\$	$s5$	3225
$\$_0 A_2 B_4$	\$	$r4$	32254
$\$_0 S_1$	\$	$r1$	322541
$\$_0 S_1$	\$	HALT	322541