Circuit Complexity

Definitions

$$P/poly$$

This is the class of languages that can be decided by a circuit family where there exists a polynomial function of the input size of the circuit that bounds the number of nodes in the circuit. That is to say that there exists $k \in N$ and $f(n) \in O(n^k)$ such that for all input sizes i, $\textbf{Size}(C_i) \leq f(i)$.

$$P\text{-Uniform } P/poly$$

This is the class of languages in $P/poly$ where a description of each circuit C_n can be generated by a polynomial-size circuit accepting 1^n as input.

$$P/O(n^k)$$

Languages that can be decided in polynomial time by a TM that accepts both the input string s with length n and an advice string a_n that depends only on n.

$$FP^{NP}$$

Class of functions $f : \{0,1\}^* \to \{0,1\}^*$ computable a polynomial time TM with access to an NP oracle.

Class Problems

Problem 1

Theorem $P \subseteq P/poly$

Proof Let T be a TM that decides $L \in P$. Let C_n be a circuit family that ignores its input and outputs 1 if the input size n is the unary encoding of a string in L or 0 otherwise. This circuit family decides the unary encoding of L and the size of each C_n is bounded by $O(n)$. Hence $L \in P/poly$. $\square$

Problem 2

Theorem $P/poly$ contains undecideable languagues.

Proof This follows from the above proof, except we allow L to be undecideable. $\square$

Problem 3

Theorem $P = P$-Uniform $P/poly$

Lemma P-Uniform $P/poly \subseteq P$

Proof Assume that a polynomial-time TM can model a polynomial-size circuit.

Then create a TM that first outputs a description of C_n from the size of the input string, and second models C_n applied to the input string. Both of these operations are polynomial-time by the above assumption, hence the entire TM is polynomial time.

Lemma $P \subseteq P$-Uniform $P/poly$

Proof: Given a TM T that decides $L \in P$, first generate a circuit family that models T via the tableau method discussed in class. The depth of each circuit C_n is $O(n^k)$ because T terminates in $O(n^k)$ steps. The width of each circuit C_n is $O(n^k)$ because T uses $O(n^k)$ tape. Therefore the size of C_n is $O(n^k)$.

Assuming the tableau method can be encoded as a polynomial-size circuit that takes the size of the string as input, the result follows. However I'm not quite sure how to prove this assumption. $\square$

Problem 4

Theorem $\bigcup_k P/O(n^k) = P/poly$

Restated, $\forall k \ . \ L \in P/O(n^k) \Rightarrow L \in P/poly$ and $L \in P/poly \Rightarrow \exists k \ . \ L \in P/O(n^k)$

Lemma $\forall k \ . \ L \in P/O(n^k) \Rightarrow L \in P/poly$

Proof Let T be a TM that accepts a string s and advice a_n. Then use the tableau method to construct a circuit family that decides whether or not $s \in L$ when given s and a_n as input. Because a_n is assumed to be known, we can hardcode each circuit with a_n as input. This new circuit family is still $O(n^k)$ in size because a_n has length bounded by $O(n^k)$. This new circuit family decides L and is polynomial in size, hence $L \in P/poly$.

Lemma $L \in P/poly \Rightarrow \exists k \ . \ L \in P/O(n^k)$

Proof Let a_n be an advice string describing the circuit C_n. Then a TM can construct C_n from this description and model it in polynomial time. Since each C_n has size bouned by $O(n^k)$, its description length is also $O(n^k)$ and so a_n is also polynomial in length. So this TM decides L in polynomial time when given a polynomial-size advice string, hence $L \in P/O(n^k)$. $\square$

Problem 5

Part a SPARSE $\subseteq P/poly$

Proof Let C_n be a circuit family that uses an encoding of each string in L of length n. C_n checks its input against each such string and therefore this circuit family decides L. Since, by the definition of sparsity, there are at most polynomially many strings with length less that n, C_n has polynomial size. Hence C_n is in $P/poly$. $\square$

Part b $P/poly = P^{\text{SPARSE}}$

Proof Let C_n be a circuit family that decides $L \in P/poly$. Then let there be a language that accepts the following data, encoded in an unambigious way:

1. An encoding of the size n of a string in L
2. The index i of a desired bit in a description of C_n

A string as described above is in the language if i-th bit of a binary diescription of C_n is 1 and not in the language if it is 0.

This language is sparse because n can be encoded in $O(\log n)$ symbols and the description of C_n uses polynomial space by definition of $P/poly$.

Assume an oracle that decides this language. Then a TM can make polynomially-many requests to this oracle to get the entire description and then simulate the circuit in polynomial time. Hence the TM runs in polynomial time and therefore $L \in P^{\text{SPARSE}}$.

If we start with a langauge in P^{SPARSE}, then we can generate a circuit family with a lookup table for the sparse language. Sparsity ensures that size of each ciruit is polynomial. $\square$

Problem 6

Theorem There are undecideable languages in $P/O(\log n)$.

Proof I will use Matthew's idea and prove a stronger result that there are undecideable languages in $P/1$.

Let L be a unary encoding of an undecideable language. Let a_n be a sequence of advice strings where $a_n = 1$ if the unary encoding of n is in L. Then a TM with access to a_n need only check the advice string corresponding to the size of the its input. So this TM runs in polynomial time and each a_n has length exactly 1. $\square$

Problem 7

Part a Provide some total ordering of assignments. This list has length $O(2^{|\phi|})$. Then make use of an efficient search algorithm like binary search:

First provide the entire list of assignemnts to the oracle. If it produces 0, then halt and return $\perp$. If it produces 1, then split the list in half. Provide the first half of the list to the oracle. If the oracle returns 0, then we know that no assignment in that half satisfies the formula and therefore the second half must contain the satisfying formula. If it returns 1, then split the first half in half again. Repeat until the satisfying assignment is found.

This algorithm completes in $O(\log n)$ time, where $n \in O(2^{|\phi|})$. So the time complexity of a TM implementing this algorithm that outputs the satisfying assignment or $\perp$ is $O(|\phi|) \in FP$. $\square$

Part b One direction is trivial because $P \subseteq P/O(\log n)$.

The other way, if $NP \subseteq P/O(\log n)$, then for each problem in NP there exists a sequence of advice strings each with length $O(\log n)$ such that a TM with access to this advice solves each instance of the problem in polynomial time.

I am not sure how this implies that $NP \subseteq P$. P is a strict subset of $P/O(\log n)$, so this does not provide path forward. I would have to make use of the fact that solutions to problems NP can be verified by languages in P. The advice string would have to tell me something about the trace of the P-time TM, however this depends on more than just the input length.

Part c This would depend on insight from part b that I don't have.

Problem 8

This goes back to the same construction used in problems 1, 2, and 6. The advice string just encodes whether or not the unary representation is in the language. This has no dependence on whether or not the lanuage is computable.