CSE547 Class 24

Jeremy Buhler

April 24, 2017

1 Complements of Languages

- For any language L, there is a complementary language $\overline{L} = \Sigma^* - L$.
- How hard is it to decide $\overline{L}$?
- We saw previously that, if L is RE but undecidable, then $\overline{L}$ is not even RE.
- However, if L is decidable, so is $\overline{L}$.
- Let's consider similar questions around the complexity of $\overline{L}$.
- If $L \in P$, then surely $\overline{L}$ is also in P!
- (Just exchange the accept and reject states of the TM deciding L.)
- What if $L \in NP$?

Let's consider an example.

- Consider SAT, the language of satisfiable Boolean formulas.
- Technically, its complement is "all strings that are not satisfiable Boolean formulas."
- But it's easy to recognize a reasonable encoding of a valid Boolean formula in polynomial time.
- So we might as well consider the language "all valid Boolean formulas that are not satisfiable" (intersection of $\overline{\text{SAT}}$ with "valid Boolean formulas").
- Call this language UNSAT.
- The "obvious" way to decide if a formula ϕ is in UNSAT is to try all possible truth assignments to ϕ and see if any of them satisfy it.
- Note that trying all possibilities is *different* from nondeterminism!
- An NTM accepts w if $\exists$ a computation on w that accepts.
- However, an NTM cannot reject if $\exists$ a computation that rejects w, and it cannot "run on w, then make the opposite decision."
- It's not clear if there exists an NTM that can decide UNSAT.

- Put another way, it's not clear if there exists a short certificate that allows you to verify that a formula is unsatisfiable.

Hmmm.... time for a new complexity class.

- Let CoNP be the set of all languages L such that $\overline{L} \in NP$.
- Every language in P is in both NP and coNP, since "coP" is the same as P – given a deterministic polytime decider M for L, just flip its accepting and rejecting states.
- But what about problems in CoNP that are not obviously in P?
- UNSAT is an example of such a problem.
- Might we be able to simulate "try all possibilities" with nondeterminism?
- Put another way, is NP the same as coNP?

As of now, we don't know! Here's one observation...

- *Lemma*: if any NP-complete language is in CoNP, then NP = CoNP.
- **Pf**: let L be an NP-complete language, and suppose that L is in CoNP.
- Then $\overline{L} \in NP$.
- For any $L' \in NP$, $L' \leq_p L$ by some polytime transducer f.
- But using the same transducer, we have that $\overline{L'} \leq_p \overline{L}$.
- Conclude that $\overline{L'} \in NP$ too, since it polynomially reduces to a problem in NP.
- Hence, the complement of every problem in NP is also in NP. QED

The NP = coNP question is not as well-known as P = NP, but it is equally perplexing.

2 ∃- and ∀-TMs

- A nondeterministic TM chooses among multiple options for each of its moves.
- It accepts its input iff there exists a set of choices that allows it to accept.
- Hence, we might call an NTM an "∃-TM".

What's the equivalent TM for problems like UNSAT?

- Imagine a TM that accepts an input w iff *all* possible computations on w accept.
- (As for nondeterminism, all the computations are assumed to occur simultaneously.)
- This TM also has moves that involve choices, but it must enumerate *all* choices for each such move and accept only if all of the choices lead to acceptance.
- We can call such a TM a "∀-TM".

- **Lemma**: The set of problems solvable in polynomial time by a ∀-TM is exactly coNP.
- If L is in NP, then it has a polytime NTM N.
- Consider a ∀-TM N' that modifies N as follows.
  - Exchange the accepting and rejecting state of N.
  - Everywhere N chooses nondeterministically, N' enumerates all possible choices.
- N' accepts w iff every one of its computations on w accepts.
- But this implies that every computation of N on w rejects!
- Hence, N' accepts $\overline{L}$.
- Conversely, if N' is a ∀-TM accepting L in polynomial time, a similar construction creates an NTM that accepts $\overline{L}$, and so $\overline{L} \in NP$.

3 Alternating TMs

- We can invent more complex types of TM by mixing ∃- and ∀-type steps.
- Consider a class of TMs that can make both ∃- and ∀-style choices at various points in their computations.
- Depending on the current state $q \in Q$ of such a TM, it either chooses a move nondeterministically (∃ state) and accepts if the computation for at least one choice accepts, or it enumerates all possible moves (∀ state) and accepts if the computations for all choices accept.
- These TMs are called *alternating TMs* (ATMs).
- **Example**: Let MINFORM be the language of Boolean formulas of *minimum size*.
- That is, MINFORM contains all ϕ such that there is no logically equivalent but shorter formula.
- It's not known if MINFORM is in either NP or coNP.
- However, we can easily write an ATM for MINFORM that uses some ∀ steps over a sub-computation with some ∃ steps.
- On input ϕ, ∀ formulas ψ shorter than ϕ...
- If ∃ a truth assignment A such that...
- $\phi \mid_A \neq \psi \mid_A$...
- ... then accept.
- (That is, we accept ϕ iff no formula shorter than ϕ yields the same result under all possible truth assignments.)

We can define complexity classes around alternating TMs.

- **Defn**: Let ATIME($t(n)$) be the set of all languages decidable in time $O(t(n))$ by an alternating TM.
- Let
$$\text{AP} = \bigcup_{k>0} \text{ATIME}(n^k).$$
- Can we say anything useful about alternating complexity classes?
- **Thm**: for any $f(n) \geq n$,
$$\text{ATIME}(f(n)) \subseteq \text{DSPACE}(f(n)) \subseteq \text{NSPACE}(f(n)) \subseteq \text{ATIME}(f^2(n)).$$
- This implies that AP = PSPACE.
- **Pf**: For the first inclusion, we extend our deterministic simulation of nondeterministic TMs to work with general alternating TMs.
- The simulation runs "depth-first", exploring all possible computation paths, each of which must terminate after at most $O(f(n))$ steps and so touch at most $O(f(n))$ cells.
- The difference is that, when we reach a $\forall$ state, we "and" together the results of all computations reachable from the current configuration instead of stopping at the first acceptance.
- Our string on tape 2 to track the current choice at each $\exists$ or $\forall$ state can only grow as large as $O(f(n))$, the number of moves in any one computation.
- Hence, ATIME($f(n)$) $\subseteq$ DSPACE($f(n)$). QED
- For the last inclusion, let N be an NTM deciding L in space $O(f(n))$.
- Suppose N has a unique accepting configuration c_a and runs in time at most $2^{df(n)}$ on inputs of size n.
- As we saw, N accepts w in space $O(f(n))$ iff $\text{YIELDN}^{f(n)}(c_0(w), c_a, 2^{df(n)})$ is true.
- Moreover, we saw that $\text{YIELDN}^i(c, c', t)$ is true iff there exists c_m of size at most i s.t. for all pairs $(c_1, c_2) \in \{(c, c_m), (c_m, c')\}$, $\text{YIELDN}^i(c_1, c_2, t/2)$ is true.
- We will build an alternating TM to evaluate YIELDN^i for running times up to $2^{O(i)}$, in time $O(i^2)$.
- Divide the ATM's operation into $O(i)$ *levels*, each containing one $\exists$ phase over one $\forall$ phase.
- One level corresponds to one step of the recursion for YIELDN.
- The $\exists$ phase chooses the intermediate configuration c_m, while its nested $\forall$ phase enumerates all configuration pairs c_1 and c_2 and trivially accepts all choices but the pairs (c, c_m) and (c_m, c'), for which we recur explicitly.

- For $t = 1$, YIELDN directly checks that $c = c'$ or $c \vdash c'$ in time $O(i)$, requiring neither choosing nor enumerating behavior.
- Each level chooses $O(i)$ symbols to pick c_m, and for each choice enumerates $O(i)$ symbols to pick c_1 and c_2.
- Hence, the total number of choice/enumeration moves over the all $O(i)$ levels is only $O(i^2)$, while the base case work is $O(i)$.
- Conclude the ATM computes YIELDN^i for any TM in time $O(i^2)$.
- As before, we want our ATM to answer correctly for input w without having to know/compute $f(|w|)$.
- Given w, the ATM first nondeterministically *guesses* the value $f(|w|)$, then computes as described above whether w can be accepted in this much space.
- (Think of the ATM starting with guess $i = 1$ and incrementing this guess nondeterministically until it decides to stop and compute YIELDN^i.)
- If N accepts w, it does so in space $O(f(|w|))$; hence, the ATM will guess the space bound correctly given time $O(f(|w|))$ and hence will find an accepting computation for w in time $O(f^2(|w|))$.
- If N does not accept w, no guess will work, and so the ATM will reject w.
- Conclude that NSPACE$(f(n)) \subseteq ATIME(f^2(n))$. QED