CS 221: Computational Complexity **Prof. Salil Vadhan**

Problem Set 2

Assigned: Wed. Feb. 17, 2010 Due: Thu. Mar. 4, 2010 (5 PM sharp)

- You must *type* your solutions. LaTeX, Microsoft Word, and plain ascii are all acceptable. Submit your solutions *via email* to `cs221-hw@seas.harvard.edu`. If you use LaTeX, please submit both the compiled file (`.ps`) and the source (`.tex`). Please name your files `PS2-yourlastname.*`.
- Strive for clarity and conciseness in your solutions, emphasizing the main ideas over low-level details. Do not despair if you cannot solve all the problems! Difficult problems are included to stimulate your thinking and for your enjoyment, not to overwork you. We'll try to mark harder problems with a *.

Problem 1. (Certificate characterization of NL?) It is tempting to try to characterize **NL** as follows. We say a language L has *logspace-verifiable certificates* if there is a logspace algorithm M and a polynomial p such that $x \in L$ if and only if there exists a string y of length at most $p(|x|)$ such that $M(x, y) = 1$.

Prove that the class of languages with logspace-verifiable certificates is exactly **NP** (and hence is unlikely to equal **NL**). As mentioned in class and shown in the text, if we restrict the verifier to have *one-way access* to the certificate, then we obtain the class **NL**.

Solution.

- The class of languages with logspace-verifiable certificates is **NP**.
  Let the class of languages with logspace-verifiable certificates be **C**. First, since every logspace algorithm performs in polynomial time, $\mathbf{C} \subseteq \mathbf{NP}$.

  Then consider $L \in \mathbf{NP}$, $\forall$ input $x \in \{0,1\}^*$, $\exists$ polynomial-sized CNF ϕ_x such that $x \in L$ iff $\exists y, \phi_x(y) = 1$. Now, we can construct a logspace algorithm A to check whether the certificate y satisfies ϕ_x clause by clause. As a result, $\mathbf{NP} \subseteq \mathbf{C}$.

  Finally, we have $\mathbf{C} = \mathbf{NP}$.
- Once we restrict the verifier to have *one-way access* to the certificate, then we have $\mathbf{C} = \mathbf{NL}$.
  First, since PATH is **NL**-complete, $\forall L \in \mathbf{NL}$ and input $x \in \{0,1\}^*$, $x \in L$ iff $\exists$ polynomial-sized path certificate y such that y is a valid path in the reduced graph. While we can check whether there exists an edge in the graph one by one, the certificate is only accessed once and the space required is log-sized, we have $\mathbf{NL} \subseteq \mathbf{C}$.

  Conversely, $\forall$ *logspace-verifiable* certificates can be simulated in a logspace Turing machine by its definition. That is, $\mathbf{C} \subseteq NL$.

  As a result, when we restrict the verifier to have only *one-way access*, we obtain $\mathbf{C} = \mathbf{NL}$.

Problem 2. (NL-completeness) Prove that 2SAT is **NL**-complete. (Hint: To prove that it is in **NL**, show that the satisfiability of ϕ can be determined from the answers to polynomially many PATH questions involving the directed graph G_ϕ that includes edges $(\neg x, y)$ and $(y, \neg x)$ for every clause $(x \vee y)$ in ϕ.)

Solution. Before the following proof, I want to emphasize on an important observation: for any satisfiable 2SAT ϕ, and a clause $(a \vee b)$ in ϕ. If a is *False*, then b must be *True*.

(2SAT $\in$ NL) Consider arbitrary $\phi \in s$SAT, construct a corresponding graph $G_\phi = \{V, E\}$ as follow:

- V = {All the literals in ϕ including their negations}.
- E = $\{(\neg a, b), (\neg b, a) : (a \vee b) \in \phi\}$.

First, we claim that $\phi \in$ 2SAT iff the graph G_ϕ has a cycle that contains both a variable and its negation, then show that the transformation and verification in G_ϕ can be done in logspace.

We start from the observation mentioned in the beginning of the whole proof. Consider a clause $(a \vee b) \in \phi$ and the corresponding edge $(\neg a, b) \in G_\phi$, we have the fact that if $\neg a$ is *True* (i.e. a is *False*) then b is also *True*. As a result, we can see that all the literals in a cycle of G_ϕ are simultaneously *True* or *False* since once the *True* assigned to one of them, it will transfer to everyone.

Claim 1 $\phi \in$ 2SAT $\Leftrightarrow$ *There is no cycle of G_ϕ containing a literal and its negation*

Proof of claim:

1. If $\phi \in$ 2SAT and suppose there is a cycle of G_ϕ containing both variable a and its negation $\neg a$. With the above observation, we know that a and $\neg a$ are the same, which comes to a contradiction. Thus, there's no cycle containing both a literal and its negation.
2. Conversely, if G_ϕ has no cycle containing both a literal a and its negation $\neg a$, the following procedure construct a satisfying assignment: Start from the first clause of ϕ, $(a \vee b)$. If there is a path from a to $\neg a$, then assign a to be *False*, and the same for b. Now you must wonder whether there will exist paths from a to $\neg a$ and from b to $\neg b$? The fact is No. Since $(a \vee b) \in \phi$, we have $(\neg a, b), (\neg b, a) \in E$. As a result, there will be a cycle containing both a and $\neg a$, which contradicts to the assumption.

□

Finally, we can see that to verify whether there exists a cycle containing both a literal and its negation, a simple PATH algorithm can be applied. Concretely, we check whether there is a path from a literal a to its negation $\neg a$ for all literal. This procedure can be done in logspace with the knowledge that PATH$\in$ **NL**. As a result, we have 2SAT $\in$ **NL**.

(NL $\leq_l$ 2SAT) Since we know that PATH is **NL**-complete, to show the **NL**-completeness of 2SAT, it's sufficient to reduce PATH to 2SAT in logspace.

Given $(G = (V, E), s, t) \in$ PATH where G is a directed graph and $s, t \in V$. The decisional problem is to decide whether there exists a path from s to t.

First, let's construct a 2SAT $\phi_{G,s,t}$ in the following way:

- Tag each vertex in V with different literal, say $v_1, ..., v_n$, where $n = |V|$. WLOG, tag s with v_1 and t with v_n.
- If there is a path from vertex v_i to v_j, then add a clause $(\neg v_i \vee v_j)$ into $\phi_{G,s,t}$.
- Put additional clause $(v_1 \vee x)$, $(v_1 \vee \neg x)$, $(\neg v_n \vee y)$, and $(\neg v_n \vee \neg y)$ into $\phi_{G,s,t}$.

Know we claim that there is a path from s to t in G iff $\phi_{G,s,t}$ is **not** satisfiable.

Claim 2 *$(G = (V, E), s, t) \in$ PATH and $\phi_{G,s,t}$ is the corresponding 2SAT by the construction mentioned before. Then, there is a path from s to t in G iff $\phi_{G,s,t}$ is* **not** *satisfiable.*

Proof of claim: The claim is equivalent to the following statement:

$$\phi_{G,s,t} \; is \; satisfiable \; \Leftrightarrow \; \nexists s \rightsquigarrow t \in G$$

1. ($\phi_{G,s,t}$ is satisfiable$\Rightarrow \nexists s \rightsquigarrow t \in G$)
   Suppose $\phi_{G,s,t}$ is satisfiable, then v_1 must be assigned to *True* since the clauses $(v_1 \vee x)$ and $(v_1 \vee \neg x)$ are in $\phi_{G,s,t}$. And v_n must be assigned to *False* for the same reason. However, once there is a path from s to t, we know that the literal correspond to each vertex on the path must be assigned to *True* as long as the literal v_1 corresponding to the first vertex s is assigned to *True*. (By the observation mentioned in the (2SAT $\in$ **NL** proof)). As a result, v_n should be assigned *True* and simultaneously, which comes to a contradiction.
2. ($\nexists s \rightsquigarrow t \in G \Rightarrow \phi_{G,s,t}$ is satisfiable)
   Suppose there is no path from s to t in G. Then we can construct a satisfying assignment for $\phi_{G,s,t}$. First, set v_1 to be *True* and v_n to be *False*. Then extend the assignment to other related literals. Since there exists no path from s to t, the extension from v_1 outward will not affect the assignment of v_n and its extension. After that, assign rest of the literals without making conflict. This is possible since there's no literal and its negation on the same path in G. As a result, when there is no path from s to t, then $\phi_{G,s,t}$ is satisfiable.

□

Finally, we conclude that 2SAT is **NL**-complete.

Problem 3. (A PSPACE-complete game) In the World Domination game, two players are given an undirected graph of initially unoccupied countries, a score for each country, and an initial score for each player. They take turns to take up an unoccupied country, provided that it is not adjacent to the other player's existing territory (countries), to avoid frictions. Each country is associated with a score, and the player with higher total score wins.

Show that deciding whether the first player has a winning strategy in the World Domination game (given the country graph and scores in binary) is **PSPACE**-complete.

Solution. First, let's define the decisional World Domination game more formally:

Definition 3 *$(G = (V, E, S), M_A, M_B)$ is a World Domination game, where G is an undirected graph with vertex set V, edge set E and a score vector S recording the score of each vertex. M_A and M_B are initial score for player A and B respectively. Define the decisional World Domination game (*DECISION-WDG*) as*

$$\text{DECISION-WDG} = \{(G = (V, E, S), M_A, M_B) : \exists \textit{ a strategy for } A \textit{ to win the game}\}$$

Claim 4 *With the definition above, we claim the following two things:*

1. DECISION-WDG$\in$ **PSPACE**
2. **PSPACE** $\leq_p$DECISION-WDG

Proof of claim:

1. For any $(G = (V, E, S), M_A, M_B)$, we can directly encode the game in polynomial-sized space. Furthermore, we can try every possible strategy of A be recording the choices they make. Suppose the size of the problem is n, we can see that there are at most n rounds of the game and we can store each choice in $\log n$ space. Thus, we can simulate all the possible strategies in $poly(n)$ spaces.
2.

□

Problem 4. (Collapsing the Hierarchy) Show that $\mathbf{\Sigma}_k^p = \mathbf{\Pi}_k^p$ implies $\mathbf{PH} = \mathbf{\Sigma}_k^p$, i.e. the polynomial hierarchy collapses to the kth level.

Solution. Suppose that $\Sigma_k^p = \prod_k^p = \Sigma_i^p$, the following will show $\Sigma_{i+1}^p = \Sigma_k^p$.

Since we know that $\Sigma_{i+1}^p = \mathbf{NP}^{\Sigma_i^p}$, $\forall L \in \Sigma_{i+1}^p$ and input $x \in \{0,1\}^*$, if $x \in L$, $\exists$ a certificate y and polynomially many Σ_i^p queries $q_1, ..., q_t$ with answers $a_1, ..., a_t$ such that $|y|, t = poly(|x|)$. Thus, we can create a Σ_i^p predicate p to verify that $p(x, y, q_1, ..., q_t, a_1, ..., a_t) = 1$ iff $x \in L$. As long as the number of queries are polynomially many, we conclude that $L \in \Sigma_i^p = \Sigma_k^p$.

With induction, we know that $\forall i \geq k$, $\Sigma_i^p = \Sigma_k^p$. By the definition of polynomial hierarchy, we have $\mathbf{PH} = \Sigma_k^p$.

Problem 5. (More Time-Space Tradeoffs for SATISFIABILITY**)** The time-space tradeoffs done in class optimize the space lower bound $(n^{1-\epsilon})$ while giving a relatively weak time lower bound $(n^{1+o(1)})$. On this problem, you'll do the opposite, giving a time lower bound of $n^{1.41}$ while giving a weaker space lower bound $(n^{o(1)})$.

Do not worry about constructibility of the time and space bounds on this problem.

1. Show that for every $T(n) \geq n^2$, $\mathbf{TISP}(T, T^{o(1)}) \subseteq \Sigma_2\mathbf{TIME}(T^{1/2+o(1)})$.
2. Use the above to prove that SAT $\notin$ $\mathbf{TISP}(n^c, n^{o(1)})$ for any $c < \sqrt{2}$. (Hint: Use a NONdeterministic-time Hierarchy Theorem.)

Solution.

1. Given $L \in \mathbf{TISP}(T, T^{o(1)})$ and input $x \in \{0,1\}^*$, $\exists$ a Turing machine$M$ fed with x that runs in time $O(T(|x|))$ and uses space $O(T^{o(1)}(|x|))$. Suppose $|x| = n$, consider the configuration graph $G_{M,x}$ of M on input x. By definition, each configuration graph can be described in $O(T^{o(1)}(n))$ space.

   Moreover, $x \in L$ iff there exists a path in $G_{M,x}$: $C_1, ..., C_{T^{\frac{1}{2}}(n)}$, where $C_0 = C_{start}$ is the starting configuration and $C_{T^{\frac{1}{2}}(n)}$ is one of the accepting configuration. Formally writing,

   $$x \in L \Leftrightarrow \exists C_1, ..., C_{T^{\frac{1}{2}}(n)}, C_0 = C_{start}, C_{T^{\frac{1}{2}}(n)} = C_{accept}, C_i \ reaches \ C_{i+1} \forall 0 \leq i \leq T^{\frac{1}{2}}(n) - 1$$

   Since checking C_i reaches C_{i+1} takes $O(T^{\frac{1}{2}+o(1)}(n))$ and finding the existence path takes $O(T^{\frac{1}{2}})$, we can find out that the whole process is in $\Sigma_2\mathbf{TIME}(T^{\frac{1}{2}+o(1)}(n))$. As a result, $\mathbf{TISP}(T, T^{o(1)}) \in \Sigma_2\mathbf{TIME}(T^{\frac{1}{2}+o(1)})$.

   Remark 5 The reason why assuming $T(n) \geq n^2$ is because that we need to have $T^{\frac{1}{2}}(n)$ configurations in the path. While $T(n) < n^2$, this we not make sense.

2. To show the correctness of the statement, we first claim that if SAT $\in \mathbf{TISP}(n^c, n^{o(1)})$, where $c < \sqrt{2}$, then $\mathbf{NTIME}(n) \in \mathbf{TISP}(n^{c+o(1)}, n^{o(1)}) \subseteq \mathbf{DTIME}(n^c)$.

   Second, with this claim and the result from Item 1 we have $\mathbf{TISP}(n^{c+o(1)}, n^{o(1)}) \subseteq \Sigma_2\mathbf{TIME}(n^{\frac{c}{2}+o(1)}) \subseteq \mathbf{NTIME}(n^{\frac{c^2}{2}+o(1)}) \subset \mathbf{NTIME}(n)$, which leads to a contradiction: $\mathbf{NTIME}(n) \subseteq \mathbf{NTIME}(n)$ to nondeterministic-time hierarchy theorem. Thus, SAT $\notin \mathbf{TISP}(n^c, n^{o(1)})$.

   Claim 6 *If* SAT $\in \mathbf{TISP}(n^c, n^{o(1)})$*, where* $c < \sqrt{2}$*, then* $\mathbf{NTIME}(n) \subseteq \mathbf{TISP}(n^{c+o(1)}, n^{o(1)})$.

   Proof of claim: By Cook-Levin theorem, we know that if SAT $\in \mathbf{TISP}(n^c, n)$, where $c < \sqrt{2}$, $\mathbf{NTIME}(n) \subseteq \mathbf{TISP}(n^c \text{polylog}(n), n^{o(1)}\text{polylog}(n)) \subseteq \mathbf{TISP}(n^{\sqrt{c}+o(1)}, n^{o(1)})$. □

   Corollary 7 *If* SAT $\in \mathbf{TISP}(n^c, n^{o(1)})$*, where* $c < \sqrt{2}$*, then* $\mathbf{NTIME}(n) \subseteq \mathbf{DTIME}(n^{c+o(1)})$

   Claim 8 *If* $\mathbf{NTIME}(n) \subseteq \mathbf{DTIME}(n^{c+o(1)})$*, then* $\Sigma_2\mathbf{TIME}(n^{\frac{c}{2}+o(1)}) \subseteq \mathbf{NTIME}(n^{\frac{c^2}{2}+o(1)})$

   Proof of claim: Suppose language $L \in \Sigma_2(n^{\frac{c}{2}+o(1)})$, then

   $$x \in L \Leftrightarrow \exists y \forall z,] s.t. \ f(x, y, z) = 1$$

   where $|y|, |z| \leq p(|x|)$, $p(n) = n^{\frac{c}{2}+o(1)}$ and f can be checked in time $|x|^{\frac{c}{2}+o(1)}$. Since we assume $\mathbf{NTIME}(n) \subseteq \mathbf{DTIME}(n^{c+o(1)})$, given x, y we can verify whether $\forall z \ s.t. \ f(x, y, z) = 1$ in time $|x|^{c+o(1)}$. As a result, there is an $n^{\frac{c^2}{2}+o(1)}$ polynomial g such that

   $$x \in L \Leftrightarrow \exists y \ s.t. \ g(x, y) = 1$$

   That is, $\Sigma_2\mathbf{TIME}(n^{\frac{c}{2}+o(1)}) \subseteq \mathbf{NTIME}(n^{\frac{c^2}{2}+o(1)})$ □

Finally, sum up the above results, if SAT $\in$ **TISP**$(n^c, n^{o(1)})$, then

$$\mathbf{NTIME}(n) \subseteq \Sigma_2\mathbf{TIME}(n^{\frac{c}{2}+o(1)}) \subseteq \mathbf{NTIME}(n^{\frac{c^2}{2}+o(1)}) \subset \mathbf{NTIME}(n)$$

By nondeterministic-time hierarchy theorem, it's a contradiction to have **NTIME**$(n) \subset$ **NTIME**(n). Thus, SAT $\notin$ **TISP**$(n^c, n^{o(1)})$

Problem 6. A PSPACE-complete game In Generalized Geography game, two players take turns to pick a city from a list of n cities, such that each city begins with the last letter of the previously chosen city. No city may be chosen twice, and the player who is unable to continue the game loses.

Show that Generalized Geography problem: {List of cities where the first player has a winning strategy} is **PSPACE**-complete.