Homework 2
Due Mon Apr 11, 5:00pm

Prof. Amit Chakrabarti
Computer Science Department
Dartmouth College

General Instructions: Same as in Homework 1.
Honor Principle: Same as in Homework 1.

3. For this problem, assume that Boolean formulas are encoded as strings over the alphabet $\{0, 1, \vee, \wedge, \neg, (,)\}$, fully parenthesized to resolve ambiguities. Note that the negation operator ($\neg$) has higher priority than the other two. The variables in a formula ϕ are represented as binary substrings of ϕ with no leading zeros. For instance, the formula

$$((x_1 \wedge \neg x_2) \vee \neg x_1 \vee (x_3 \wedge \neg x_1)) \wedge \neg x_4 \wedge (\neg x_3 \vee x_5)$$

is represented as the string

$$((1 \wedge \neg 10) \vee \neg 1 \vee (11 \wedge \neg 1)) \wedge \neg 100 \wedge (\neg 11 \vee 101)\,.$$

Define $\textsc{satisfies} = \{\langle \phi, \alpha \rangle : \phi$ is a Boolean formula and the assignment α satisfies $\phi\}$. Our proof that $\textsc{sat} \in \mathsf{NP}$ boiled down to showing that $\textsc{satisfies} \in \mathsf{P}$. Indeed, we can say that

$$\textsc{sat} \;=\; \{\langle \phi \rangle : \exists\, \alpha \text{ such that } \langle \phi, \alpha \rangle \in \textsc{satisfies}\}\,.$$

Anyhow, as we were saying, we easily have $\textsc{satisfies} \in \mathsf{P}$. Prove the stronger result that $\textsc{satisfies} \in \mathsf{L}$.

This problem is all about careful implementation, so take care to specify exactly how you use the work tape of your TM. Some naïve implementations end up requiring $\Omega(\log^2 n)$ space. [2 points]

4. Many algorithms need to perform arithmetic, and this needs to be done efficiently. When designing polynomial time algorithms, we rarely stop to think about this. But for logspace algorithms, it takes a little work to convince oneself that all is well. Therefore, it's good to solve the following problem for future reference.

Prove that addition and multiplication can be done in logspace, i.e., prove that the exist (deterministic) logspace transducers M_+ and $M_\times$ that take a pair of positive integers (represented in binary) as input and produce the sum and product (respectively) of these integers as output. [2 points]

Hint: You don't need to write out all the details of the tape-handling. Instead, rely on what we prove in class about the "composition of logspace algorithms" and solve this problem "in stages," perhaps transforming the input into a more helpful format at the first stage.