Notes on NTMSAT$\leq_p$SAT

Matvey Soloviev

October 18, 2013

Credit is owed to Prof. Anuj Dawar of the Cambridge University Computer Laboratory, from whose course on Computational Complexity taught in Cambridge during Lent 2011 I have lifted this approach with minor modifications.

NTMSAT

We can define the language

$$\text{NTMSAT} = \{\langle t, 1^m, s\rangle \quad | \quad \text{the nondeterministic Turing machine encoded by } t \text{ accepts } s \text{ within } m \text{ steps}\}.$$

This language is in NP, for given any word $\langle t, 1^m, s\rangle$ that is in the language, we can take c to be a record of the sequence of transitions the NTM encoded by t undergoes when accepting s. As we know it will take at most m steps, we know that $|c|$ is $O(m) = O(|1^m|)$ and hence $O(|\langle t, 1^m, s\rangle|)$, so of length polynomial in the length of the input; also, we can easily convince ourselves that a Turing machine could simulate the computation of t on s and verify that every transition recorded in c is indeed possible at its respective point in time, and that an accepting state is reached. Furthermore, no such c can exist for $\langle t, 1^m, s\rangle$ not in NTMSAT, because its existence would intrinsically imply membership.

This language is also NP-hard, since given any language L in NP, we know by definition that there exist some t, C and k such that L is by the NTM encoded by t in $C \cdot n^k = O(n^k)$ time, and so we can reduce L to NTMSAT by taking any string s to $\langle t, 1^{C \cdot n^k}, s\rangle$ which is certainly possible in polynomial time. Hence NTMSAT is NP-complete.

SAT

We can define the following language:

$$\text{SAT} = \{\varphi \mid \varphi \text{ is a satisfiable Boolean formula}\}.$$

This language is manifestly in NP, for we could use an assignment a of truth values to the variables in φ as its certificate of membership. Since every relevant variable occurs in φ at least once, the length of this is clearly bounded above by some polynomial in the length of φ. We can also specify a (deterministic) Turing machine to quickly evaluate φ given an assignment of truth values to its variables.

We will now exhibit a polynomial reduction from NTMSAT to SAT.

NTMSAT$\leq_p$SAT

In order to specify a polynomial reduction from NTMSAT to SAT, we need to

- present an algorithm to reduce a candidate string $\langle t, 1^m, s\rangle$ to a Boolean formula φ, such that
- φ is in SAT if and only if $\langle t, 1^m, s\rangle$ is in NTMSAT and
- φ is of a length bounded above by a polynomial in the length of $\langle t, 1^m, s\rangle$ and the algorithm runs in polynomial time.

We will achieve this by expressing the statement "the nondeterministic Turing machine encoded by t accepts s in at most m steps" as a Boolean formula. To this end, we will encode the configuration of the TM $\langle Q, \Sigma \cup \{\square\}, \delta, q_0, q_{\text{acc}}\rangle$ at every point in time using Boolean variables as follows (note that in m steps, we can not visit more than m tape positions):

$S_{i,q}$	for all $1 \leq i \leq m$, $q \in Q$	iff the state at step i is q
$T_{i,j,\sigma}$	for all $1 \leq i, j \leq m$, $\sigma \in \Sigma \cup \{\square\}$	iff the tape contains symbol σ at position j at step i
$H_{i,j}$	for all $1 \leq i, j \leq m$	iff the head is at position j at step i

Recall that if $S = \{s_1, \ldots, s_n\}$, then $\bigvee_{s\in S} \psi(s) = s_1 \vee \ldots \vee s_n$.

We now form a big conjunction of all the following formulae.

At time 1, we are in state q_0, the head is at the first position and the tape contains s followed by $\square$es (initial condition):

$$S_{1,q_0} \wedge H_{1,1} \wedge \bigwedge_{n\leq|s|} T_{1,n,s_n} \wedge \bigwedge_{|s|<n\leq m} T_{1,n,\square};$$

At any time, we are in exactly one state, the head is at exactly one position, and every tape slot contains exactly one symbol (consistency):

$$\bigwedge_{i\le m}\bigwedge_{q\in Q} S_{i,q} \to \bigwedge_{q\neq q'\in Q} \neg S_{i,q'},$$

$$\bigwedge_{i\le m}\bigwedge_{j\le m} H_{i,j} \to \bigwedge_{j\neq j'\le m} \neg H_{i,j'},$$

$$\bigwedge_{i\le m}\bigwedge_{j\le m}\bigwedge_{\sigma\in\Sigma\cup\{\square\}} T_{i,j,\sigma} \to \bigwedge_{\sigma\neq\sigma'\in\Sigma\cup\{\square\}} \neg T_{i,j,\sigma'};$$

At any time, if the head is not at position j, the tape at position j may not change (consistency'):

$$\bigwedge_{i\le m}\bigwedge_{j\le m} H_{i,j} \to \bigwedge_{j\neq j'\le m}\bigwedge_{\sigma\in\Sigma\cup\{\square\}} (T_{i,j',\sigma} \to T_{i+1,j',\sigma})$$

At any time, we must choose some transition that is available in the current state given the tape symbol under the head, update the tape and state and move the head accordingly (transition):

$$\bigwedge_{i\le m}\bigwedge_{j\le m}\bigwedge_{q\in Q}\bigwedge_{\sigma\in\Sigma\cup\{\square\}} \left(H_{i,j} \wedge S_{i,q} \wedge T_{i,j,\sigma} \to \bigvee_{(q',\sigma',\mathrm{dir})\in\delta(q,\sigma)} (H_{i+1,j'} \wedge S_{i+1,q'} \wedge T_{i+1,j,\sigma'}) \right),$$

where

$$j' = \begin{cases} j+1 & \text{if dir}= R \\ j & \text{if } j=1 \text{ and dir}= L \\ j-1 & \text{if dir}= L \end{cases}$$

is the updated position of the head;

Finally, at some point within the time window we are considering, we must accept the string (success):

$$\bigvee_{i\le m} S_{i,q_{\mathrm{acc}}}.$$

Note that a $\bigvee$ or $\bigwedge$ over a set with k elements generates something of size amounting to k copies of what it quantifies, so each of the formulae we are $\wedge$ing together has length at most $C \cdot m^3 \cdot |Q|^2 \cdot (|\Sigma|+1)^2 \cdot |\delta(x)|$ for some constant C, of which m^3 is the only term that depends on the length of the input. So the formula we obtain is certainly of length polynomial in $m = |1^m|$, and the above construction should be straightforward to execute in polynomial time. Also, we constructed the formulae such that a satisfying assignment of truth values to the variables will give us an exact record of a computational path through the Turing machine encoded by t on s which accepts within m steps, so deciding whether such an assignment exists is tantamount to deciding whether

$\langle t, 1^m, s \rangle$ is in NTMSAT. We have therefore specified a polynomial reduction from NTMSAT to SAT; SAT is therefore NP-complete – it is in NP and a polynomial-time algorithm for SAT would imply a polynomial-time algorithm for deciding membership of NTMSAT, which would imply a polynomial-time algorithm for deciding membership of any language in NP. □