Toda's Theorem

This material was presented in class on November 15, 2012. We wish to prove

Toda's Theorem: $\mathrm{PH} \subseteq \mathrm{P}^{\#\mathrm{SAT}[1]}$. That is, for any language $L \in \mathrm{PH}$, there is a polynomial-time oracle Turing Machine that decides membership in L when given access to a #SAT oracle; moreover, on any input x, the oracle machine makes just one #SAT query.

We use the following lemmas from Chapter 17 of Arora-Barak.

Lemma 17.17: For any constant $c \in \mathcal{N}$, there exists a probabilistic polynomial-time algorithm f such that for any m and any $\sum_c$ SAT instance ψ,

$$\begin{aligned} \psi \text{ is true } &\longrightarrow \; Pr[f(\psi) \in \oplus SAT] \geq 1 - 2^{-m} \\ \psi \text{ is false } &\longrightarrow \; Pr[f(\psi) \in \oplus SAT] \leq 2^{-m} \end{aligned}$$

Lemma 17.22: There is a deterministic polynomial-time transformation T that maps CNF formulas to CNF formulas such that $\beta = T(\alpha, 1^l)$ has following property:

$$\begin{aligned} \alpha \in \oplus\mathrm{SAT} &\longrightarrow \; \#(\beta) = -1 \; (mod\ 2^{l+1}) \\ \alpha \notin \oplus\mathrm{SAT} &\longrightarrow \; \#(\beta) = 0 \; (mod\ 2^{l+1}) \end{aligned}$$

A proof of Lemma 17.17 was presented in class on November 13, 2012 and can be found on the course website. A proof of Lemma 17.22 is presented below. We now show how to use them to prove Toda's Theorem.

Note first that it suffices to reduce membership in Σ_cSAT to a single #SAT query, for an arbitrary $c \geq 1$, because every L in the PH is in Σ_c^P for some c and hence many-to-one reducible to Σ_cSAT.

Consider the probabilistic polynomial-time algorithm f in the Lemma 17.17 with $m = 2$. Instead of treating f as a probabilistic algorithm, we can treat it as a deterministic function of two arguments, namely the Σ_cSAT instance ψ and the random string r. Let $R = |r|$, and $l = R + 1$, and consider the formula,

$$\sum_{r \in \{0,1\}^k} \#T(f(\psi, r), 1^l) \qquad\qquad (*)$$

If ψ is $true$, then at least $\frac{3}{4}$ of the terms being summed in $(*)$ are -1 $mod\ 2^{l+1}$, and the rest are 0 $mod\ 2^{l+1}$. Thus, when ψ is $true$, $(*)$ falls into the interval $[-2^R, -\lceil \frac{3}{4} \times 2^R \rceil]$ $mod\ 2^{l+1}$.

If ψ is $false$, then at least $\frac{3}{4}$ of the terms being summed in $(*)$ are 0 $mod\ 2^{l+1}$, and the rest are -1 $mod\ 2^{l+1}$. Thus, $(*)$ falls into the interval $[-\lceil \frac{1}{4} \times 2^R \rceil, 0]$ $mod\ 2^{l+1}$ in this case.

Because $2^{l+1} > 2^{R+1}$, the two intervals in these two cases are disjoint. Hence, if we can show how to compute $(*)$ in $\mathrm{P}^{\#\mathrm{SAT}[1]}$, we can decide which of the two intervals it falls into to get the truth value of ψ.

Note that $\beta = T(f(\psi, r), 1^l)$ is a SAT instance. Thus, we can apply the parsimonious Cook-Levin reduction to the nondeterministic, polynomial-time Turing Machine that takes (ψ, r) as input and accepts if and only there exists a witness y of length polynomial in the input size that satisfies β. Call the output of that reduction $\Gamma(\psi, r, y, z)$. (The string z represents the extra variables used in the Cook-Levin reduction to encode the sequence of snapshots.) Let $\Gamma_\psi(r, y, z)$ denote $\Gamma(\psi, r, y, z)$ for a fixed formula ψ, and let CL denote the (polynomial-time computable, many-to-one) reduction function. Then,

$\#\Gamma_\psi(r,y,z)$
$= |\{(r,y,z) \mid (y,z) \textit{ satisfies } CL(T(f(\psi,r),1^l))\}|$
$= |\{(r,y,z) \mid (y,z) \textit{ satisfies } CL(T(f(\psi,r),1^{|r|+2}))\}|$
$= \sum_{r\in\{0,1\}^R}$ *the number of* (y,z) *pairs that satisfy* $T(f(\psi,r),1^{|r|+2})$
(*because the reduction is parsimonious*)
$= \sum_{r\in\{0,1\}^R} \#T(f(\psi,r),1^{|r|+2})$
$= (*)$

Thus, given ψ and r, we can first compute the value of β, apply the parsimonious Cook-Levin reduction to it to obtain $\Gamma_\psi(r,y,z)$, then get the value of $(*)$ by making one query to the #SAT oracle.

Proof of Lemma 17.22: Recall that we have defined addition and multiplication operators on CNF formulas with the properties that $\#(\phi+\tau) = \#(\phi)+\#(\tau)$ and $\#(\phi\cdot\tau) = \#(\phi)\cdot\#(\tau)$. (See formulas 17.5 and 17.7.) Using these operators, we can construct from any CNF formula τ a related CNF formula $4\tau^3+3\tau^4$ that, for any $i \geq 0$, satisfies

$$\#(\tau) \equiv 0 \pmod{2^{2^i}} \longrightarrow \#(4\tau^3+3\tau^4) \equiv 0 \pmod{2^{2^{i+1}}} \qquad (**)$$

and

$$\#(\tau) \equiv -1 \pmod{2^{2^i}} \longrightarrow \#(4\tau^3+3\tau^4) \equiv -1 \pmod{2^{2^{i+1}}}. \qquad (***)$$

To prove (**), let $B = \#(\tau) = C\cdot 2^{2^i}$. Then

$$\#(4\tau^3+3\tau^4) = 4B^3+3B^4 = B^2(4B+3B^2) = C^2\cdot 2^{2^{i+1}}\cdot(4B+3B^2) \equiv 0 \pmod{2^{2^{i+1}}}.$$

To prove (***), let $B = \#(\tau) = C\cdot 2^{2^i}-1$. Then

$$\begin{aligned}
\#(4\tau^3+3\tau^4) &= 4B^3+3B^4 \\
&= B^2\cdot B\cdot(4+3B) \\
&= (C\cdot 2^{2^i}-1)^2\cdot(C\cdot 2^{2^i}-1)\cdot(3C\cdot 2^{2^i}+1) \\
&= (C^2\cdot 2^{2^{i+1}}-2C\cdot 2^{2^i}+1)(3C^2\cdot 2^{2^{i+1}}-2C\cdot 2^{2^i}-1) \\
&\equiv (-2C\cdot 2^{2^i}+1)(-2C\cdot 2^{2^i}-1) \equiv -1 \pmod{2^{2^{i+1}}}
\end{aligned}$$

To get a polynomial-time transformation T with the desired property, let $\psi_0 = \alpha$, $\psi_{i+1} = 4\psi_i^3+3\psi_i^4$, and $\beta = \psi_{\lceil\log(l+1)\rceil}$.