## Technical Appendix

**Threshold DNFs** An exponentially large standard DNF is required to represent a threshold DNF. Consider a threshold DNF with just a single term containing all $n$ of the binary variables $x_1, \ldots, x_n$, and let the threshold value be $\frac{n}{2}$, with each variable having equal weight. Consider any standard DNF for this $\frac{n}{2}$ threshold function, and let $T$ be any term in it. If $T$ has fewer than $\frac{n}{2}$ variables appearing unnegated, then $T$ has a satisfying assignment with fewer than $\frac{n}{2}$ bits on, which is a contradiction. Furthermore, if $T$ has any variables appearing negated, then these variables can be removed, and all previously satisfying assignments for $T$ will remain satisfying. Thus, the smallest $T$ will contain only unnegated variables, and at least $\frac{n}{2}$ of them. The DNF formula must contain such a term for every possible subset of $\frac{n}{2}$ bits, of which there are exponentially many.

**Section 5 Assumptions** In Section 5 we assume that the tree is binary and that the distribution $P$ is given over the leaves only. To remove the first assumption, rather than summing over $\{x, y \mid x + y = i\}$, we use another dynamic program that calculates the probability that exactly $i$ amount of weight is distributed among the children of $u$. The approach is similar to calculating the probability that exactly $k$ out of $n$ biased coins come up heads. To remove the second assumption, we check at each node $u$ (not just the leaves) whether $u$ corresponds to a literal in $T$, and if so, we make a case analysis similar to the one currently restricted to our base case.

**Lemma 1.** The GenAssign subroutine for DNF formulas generates an assignment $v \in V$ with probability $\hat{P}(v)/\hat{P}(T)$.

*Proof.* Let $x_1, \ldots, x_m$ denote variables in the order as they appear in the loop from lines 5 to 12, where $m = n - |T|$. Let $v$ be the assignment generated by GenAssign$(P, T, \epsilon)$. Let $l_i = (x_i, v_{x_i})$. The probability that $v$ was generated is

$$\frac{\hat{P}(T \wedge l_1)}{\hat{P}(T)} \times \frac{\hat{P}(T \wedge l_1 \wedge l_2)}{\hat{P}(T \wedge l_1)} \times \ldots \times \frac{\hat{P}(v)}{\hat{P}(T \wedge l_1, \ldots, l_m)}.$$

After cancelling terms, we have $\hat{P}(v)/\hat{P}(T)$. □

**Lemma 2.** The GenAssign subroutine for threshold DNFs generates an assignment $v \in V$ with probability $P(v)/P(T)$.

*Proof.* The probability that an assignment $v$ is generated is:

$$\frac{\Pr[Q_r^T(i)]}{\sum_{i=q}^{W(T)} \Pr[Q_r^T(i)]} \times$$
$$\frac{\Pr[r = v_r] \Pr[Q_r^T(i) \mid r = v_r]}{\Pr[Q_r^T(i)]} \times$$
$$\frac{\Pr[Q_{r_L}^T(x) \mid r = v_r] \Pr[Q_{r_R}^T(y) \mid r = v_r]}{\Pr[Q_r^T(i) \mid r = v_r]} \times$$
$$\frac{\Pr[r_L = v_{r_L} \mid r = v_r] \Pr[Q_{r_L}^T(x) \mid r_L = v_{r_L}]}{\Pr[Q_{r_L}^T(x) \mid r = v_r]} \times$$
$$\frac{\Pr[r_R = v_{r_R} \mid r = v_r] \Pr[Q_{r_R}^T(y) \mid r_R = v_{r_R}]}{\Pr[Q_{r_R}^T(y) \mid r = v_r]} \times \ldots$$

After cancelling terms, we have

$$\frac{\Pr[r = v_r] \Pr[r_L = v_{r_L} \mid r = v_r] \Pr[r_R = v_{r_R} \mid r = v_r] \ldots}{\sum_{i=q}^{W(T)} \Pr[Q_r^T(i)]}$$
$$= \frac{\Pr[r = v_r \wedge r_L = v_{r_L} \wedge r_R = v_{r_R} \ldots]}{P(T)}$$
$$= \frac{P(v)}{P(T)}.$$

□

**Lemma 3.** The *GenAssign* subroutine for threshold DNFs generates an assignment $v \in V$ such that $v$ satisfies $T$.

*Proof.* It suffices to prove that the process generates an assignment $v$ in which the weighted sum of satisfied literals is equal to $i$, where $i \geq q$ is the value we chose in the first step. We will use induction to prove the following more general claim. For any internal node $u$, if we have chosen the value for the weighted sum of satisfied literals in $u$'s subtree to be $i$, then the generated assignment will meet this requirement.

Suppose we are at a leaf node $u$ where $l = (u, 1) \in T$ and we have chosen $u_P = b$. We then choose $i$ with probability proportional to $\Pr[Q_u^T(i)]$. The only values for $i$ which correspond to a nonzero probability are $w(l)$ and 0. In order for $u$'s literal to be satisfied, $u$'s value must be 1. So if we have chosen $i$ to be $w(l)$, then we should choose $u$'s value to be 1 with probability 1. According to the subroutine, we choose the value for $u$ to be 1 with probability equal to

$$\frac{\Pr[u = 1 \mid u_P = b] \Pr[Q_u^T(w(l)) \mid u = 1]}{\Pr[Q_u^T(w(l)) \mid u_P = b]}$$
$$= \frac{\Pr[u = 1 \mid u_P = b](1)}{\Pr[u = 1 \mid u_P = b]}$$
$$= 1.$$

On the other hand, if we have chosen $i$ to be 0, then we should choose $u$'s value to be 0 with probability 1. According to the process, we choose the value for $u$ to be 0 with probability equal to

$$\frac{\Pr[u = 0 \mid u_P = b] \Pr[Q_u^T(0) \mid u = 0]}{\Pr[Q_u^T(0) \mid u_P = b]}$$
$$= \frac{\Pr[u = 0 \mid u_P = b](1)}{\Pr[u = 0 \mid u_P = b]} = 1.$$

The case where $l = (u, 0) \in T$ follows similarly. For the inductive step, suppose we are at a node $u$ and have chosen the value $i$. According to the subroutine, we have also chosen values $x, y$ for the subtrees of $u_L$ and $u_R$, such that $x + y = i$. By the induction hypothesis, we can assume that the conditions were met for $u_L$ and $u_R$. Thus, the weighted sum of satisfied literals in the subtree of $u$ will be equal to $x + y = i$. □