We write μ_δ to denote a Bernoulli distribution, where outcome 1 occurs with probability δ and 0 with probability $1-\delta$ where $0 \leq \delta \leq 1$. Moreover, we use μ_δ^k to denote a probability distribution over k-tuples, where each bit of a k-tuple is drawn independently according to μ_δ. Finally, let $u \leftarrow \mu_\delta^k$ denote that a k-tuple u is chosen according to μ_δ^k.

The protocol execution between two probabilistic algorithms A and B is denoted by $\langle A, B\rangle$. The output of A in such a protocol execution is denoted by $\langle A, B\rangle_A$ and of B by $\langle A, B\rangle_B$. Finally, let $\langle A, B\rangle_{\text{trans}}$ denote the transcript of communication between $\langle A, B\rangle_{\text{trans}}$.

We define a *two phase algorithm* $A := (A_1, A_2)$ as an algorithm where in the first phase an algorithm A_1 is executed and in the second phase an algorithm A_2.

Definition 1.1 (Dynamic weakly verifiable puzzle.) *A dynamic weakly verifiable puzzle (DWVP) is defined by a probabilistic algorithm P called a problem poser. A problem solver $S := (S_1, S_2)$ for P is a probabilistic two phase algorithm. We write $P_n(\pi)$ to denote the execution of P with the randomness fixed to $\pi \in \{0,1\}^n$, and $(S_1, S_2)(\rho)$ to denote the execution of both S_1 and S_2 with the randomness fixed to $\rho \in \{0,1\}^*$.*

In the first phase, the poser $P_n(\pi)$ and the solver $S_1(\rho)$ interact. As the result of the interaction $P_n(\pi)$ outputs a verification circuit Γ_V and a hint circuit Γ_H. The algorithm $S_1(\rho)$ produces no output. The circuit Γ_V takes as input $q \in Q$, an answer $y \in \{0,1\}^*$, and outputs a bit. We say that an answer (q, y) is a correct solution if and only if $\Gamma_V(q, y) = 1$. The circuit Γ_H on input $q \in Q$ outputs a hint such that $\Gamma_V(q, \Gamma_H(q)) = 1$.

In the second phase, S_2 takes as input $x := \langle P_n(\pi), S_1(\rho)\rangle_{trans}$, and has oracle access to Γ_V and Γ_H. The execution of S_2 with the input x and the randomness fixed to ρ is denoted by $S_2(x, \rho)$. The queries of S_2 to Γ_V and Γ_H are called verification queries and hint queries respectively. The algorithm S_2 asks at most h hint queries, v verification queries, and succeeds if and only if it makes a verification query (q, y) such that $\Gamma_V(q, y) = 1$, and it has not previously asked for a hint query on q.

Definition 1.2 (k-wise direct-product of DWVPs.) *Let $g : \{0,1\}^k \to \{0,1\}$ be a monotone function and $P^{(1)}$ a problem poser as in Definition 1.1. The k-wise direct product of $P^{(1)}$ is a DWVP defined by a probabilistic algorithm $P^{(g)}$. We write $P_{kn}^{(g)}(\pi^{(k)})$ to denote the execution of $P^{(g)}$ with the randomness fixed to $\pi^{(k)} := (\pi_1, \ldots, \pi_k)$ where for each $1 \leq i \leq n : \pi_i \in \{0,1\}^n$. Let $(S_1, S_2)(\rho)$ be a solver for $P^{(g)}$ as in Definition 1.1. In the first phase, the algorithm $S_1(\rho)$ sequentially interacts in k rounds with $P_{kn}^{(g)}(\pi^{(k)})$. In the i-th round $S_1(\rho)$ interacts with $P_n^{(1)}(\pi_i)$, and as the result $P_{kn}^{(g)}(\pi^{(k)})$ generates circuits Γ_V^i, Γ_H^i. Finally, after k rounds $P_{kn}^{(g)}(\pi^{(k)})$ outputs a verification circuit*

$$\Gamma_V^{(g)}(q, y_1, \ldots, y_k) := g(\Gamma_V^1(q, y_1), \ldots, \Gamma_V^k(q, y_k))$$

and a hint circuit

$$\Gamma_H^{(k)}(q) := (\Gamma_H^1(q), \ldots, \Gamma_H^k(q)).$$

If it is clear form a context we omit the parameter n and write $P(\pi)$ instead of $P_n(\pi)$ where $\pi \in \{0,1\}^n$.

A verification query (q, y) of a solver S for which a hint query on this q has been asked before can not be a successful verification query. Therefore, without loss of generality, we make the assumption that S does not ask verification queries on q for which a hint query has been asked before. Furthermore, we assume that once S asked a successful verification query, it does not ask any further hint or verification queries.

Let C be a circuit that corresponds to a solver S as in Definition 1.1. Similarly as for a two phase algorithm, we write $C(\rho) := (C_1, C_2)(\rho)$ to denote that C in the first phase uses a circuit C_1 and in the second phase a circuit C_2. Additionally, the randomness in both phases is fixed to $\rho \in \{0,1\}^*$.

Experiment $Success^{P,C}(\pi, \rho)$

Oracle: A problem poser P, a solver circuit $C = (C_1, C_2)$.
Input: Bitstrings $\pi \in \{0,1\}^n$, $\rho \in \{0,1\}^*$.
Output: A bit $b \in \{0,1\}$.

Run $\langle P(\pi), C_1(\rho)\rangle$
  $(\Gamma_V, \Gamma_H) := \langle P(\pi), C_1(\rho)\rangle_P$
  $x := \langle P(\pi), C_1(\rho)\rangle_{\text{trans}}$

Run $C_2^{\Gamma_V,\Gamma_H}(x, \rho)$
  if $C_2^{\Gamma_V,\Gamma_H}(x, \rho)$ asks a verification query (q, y) such that $\Gamma_V(q, y) = 1$ **then**
    return 1
return 0

We define the *success probability* of C in solving a puzzle defined by P as

$$\Pr_{\pi,\rho}[Success^{P,C}(\pi, \rho) = 1]. \tag{0.0.1}$$

Furthermore, we say that C succeeds for π, ρ if $Success^{P,C}(\pi, \rho) = 1$.

Theorem 1.3 (Security amplification for a dynamic weakly verifiable puzzle.) *Let $P^{(1)}$ be a fixed problem poser as in Definition 1.1, and $P^{(g)}$ be a poser for the k-wise direct product of $P^{(1)}$. There exists a probabilistic algorithm Gen with oracle access to: a solver circuit C for $P^{(g)}$, a monotone function $g : \{0,1\}^k \to \{0,1\}$ and $P^{(1)}$. Additionally, Gen takes as input parameters ε, δ, the value n being the length of the input bitstring to $P^{(1)}$, the number of verification queries v and hint queries h asked by C, and outputs a solver circuit D for $P^{(1)}$ as in Definition 1.1 such that the following holds:*
If C is such that

$$\Pr_{\pi^{(k)},\rho}\left[Success^{P^{(g)},C}(\pi^{(k)}, \rho) = 1\right] \geq 8(h+v)\left(\Pr_{u \leftarrow \mu_\delta^k}[g(u) = 1] + \varepsilon\right)$$

then D satisfies almost surely

$$\Pr_{\pi,\rho}\left[Success^{P^{(1)},D}(\pi, \rho) = 1\right] \geq (\delta + \frac{\varepsilon}{6k}).$$

Additionally, D requires oracle access to g, $P^{(1)}$, C, and asks at most $\frac{6k}{\varepsilon}\log\left(\frac{6k}{\varepsilon}\right) h$ hint queries and one verification query. Finally, $Size(D) \leq Size(C) \cdot \frac{6k}{\varepsilon}$ and $Time(Gen) = poly(k, \frac{1}{\varepsilon}, n, v, h)$.

Let $hash : Q \to \{0, 1, \ldots, 2(h+v) - 1\}$, the idea is to partition Q such that the set of preimages of 0 for $hash$ contains $q \in Q$ on which C is not allowed to ask hint queries, and the first successful verification query (q, y) of C is such that $hash(q) = 0$. Therefore, if C makes a verification query (q, y) such that $hash(q) = 0$, then we know that no hint query is ever asked on this q.

We denote the i-th query of C by q_i if it is a hint query, and by (q_i, y_i) if it is a verification query. We define now an experiment $CanonicalSuccess$ in which Q is partitioned using a function $hash$. We say that a solver circuit C succeeds in the experiment $CanonicalSuccess$ if it asks a successful verification query (q_j, y_j) such that $hash(q_j) = 0$, and no hint query q_i is asked before (q_j, y_j) such that $hash(q_i) = 0$.

Experiment $CanonicalSuccess^{P,C,hash}(\pi, \rho)$

Oracle: A problem poser P, a solver circuit $C = (C_1, C_2)$,
a function $hash : Q \to \{0, \ldots, 2(h+v) - 1\}$.
Input: Bitstrings $\pi \in \{0,1\}^n$, $\rho \in \{0,1\}^*$.
Output: A bit $b \in \{0,1\}$.

Run $\langle P(\pi), C_1(\rho)\rangle$
$(\Gamma_V, \Gamma_H) := \langle P(\pi), C_1(\rho)\rangle_P$
$x := \langle P(\pi), C_1(\rho)\rangle_{\text{trans}}$

Run $C_2^{\Gamma_V, \Gamma_H}(x, \rho)$
Let (q_j, y_j) be the first verification query of C_2 such that $\Gamma_v(q_j, y_j) = 1$.
If C_2 does not succeed let (q_j, y_j) be an arbitrary verification query.

If $(\forall i < j : hash(q_i) \neq 0)$ **and** $(hash(q_j) = 0)$ **and** $(\Gamma_V(q_j, y_j) = 1)$ **then**
return 1
else
return 0

We define the *canonical success probability* of a solver C for P with respect to a function $hash$ as

$$\Pr_{\pi,\rho}[CanonicalSuccess^{P,C,hash}(\pi, \rho) = 1]. \tag{0.0.2}$$

For fixed $hash$ and a problem poser P a *canonical success* of C for π, ρ is a situation where $CanonicalSuccess^{P,C,hash}(\pi, \rho) = 1$. We show that if a solver circuit C for $P^{(g)}$ often succeeds in the experiment $Success$, then it is also often successful in the experiment $CanonicalSuccess$. Let $\mathcal{H}$ be the family of pairwise independent functions $Q \to \{0, 1, \ldots, 2(h+v) - 1\}$. We write $hash \leftarrow \mathcal{H}$ to denote that $hash$ is chosen from $\mathcal{H}$ uniformly at random. [1]

Lemma 1.4 ***(Success probability in solving a k-wise direct product of $P^{(1)}$ with respect to a function hash.)*** *For fixed P let C be a solver for P with the success probability at least γ, asking at most h hint queries and v verification queries. There exists a probabilistic algorithm* ***FindHash*** *that takes as input: parameters γ, n, the number of verification queries v and hint queries h, and has oracle access to C and P. Furthermore,* ***FindHash*** *runs in time $O((h+v)^4/\gamma^4)$, and with high probability outputs a function $hash \in \mathcal{H}$ such that the canonical success probability of C with respect to hash is at least $\frac{\gamma}{16(h+v)}$.*

Proof. We fix a problem poser P and a solver C for P in the whole proof of Lemma 1.4. For all $m, n \in \{1, \ldots, (h+v)\}$ and $k, l \in \{0, 1, \ldots, 2(h+v) - 1\}$ by the pairwise independence property of $\mathcal{H}$, we have

$$\forall q_m, q_n \in Q, q_m \neq q_n : \Pr_{hash \leftarrow \mathcal{H}}[hash(q_m) = k \mid hash(q_n) = l] = \Pr_{hash \leftarrow \mathcal{H}}[hash(q_m) = k] = \frac{1}{2(h+v)}. \tag{0.0.3}$$

Let $\mathcal{P}_{Success}$ be a set containing all (π, ρ) for which $Success^{P,C}(\pi, \rho) = 1$. We choose $hash \leftarrow \mathcal{H}$, and fix $(\pi^*, \rho^*) \in \mathcal{P}_{Success}$. We are interested in the probability over choice of $hash$ of the event

[1] It is possible to implement a random function $hash$ efficiently by for example building its function table on the fly.

$CanonicalSuccess^{P,C,hash}(\pi^*,\rho^*) = 1$. Let (q_j, y_j) denote the first query such that $\Gamma_V(q_j, y_j) = 1$. We have

$$\begin{aligned}
&\Pr_{hash\leftarrow\mathcal{H}}[CanonicalSuccess^{P,C,hash}(\pi^*,\rho^*) = 1] \\
&\quad = \Pr_{hash\leftarrow\mathcal{H}}[hash(q_j) = 0 \wedge (\forall i < j : hash(q_i) \neq 0)] \\
&\quad = \Pr_{hash\leftarrow\mathcal{H}}[\forall i < j : hash(q_i) \neq 0 \mid hash(q_j) = 0] \Pr_{hash\leftarrow\mathcal{H}}[hash(q_j) = 0] \\
&\quad \overset{(0.0.3)}{=} \frac{1}{2(h+v)}\left(1 - \Pr_{hash\leftarrow\mathcal{H}}[\exists i < j : hash(q_i) = 0 \mid hash(q_j) = 0]\right) \\
&\quad \overset{(0.0.3)}{=} \frac{1}{2(h+v)}\left(1 - \Pr_{hash\leftarrow\mathcal{H}}[\exists i < j : hash(q_i) = 0]\right) \\
&\quad \overset{(u.b)}{\geq} \frac{1}{2(h+v)}\left(1 - \sum_{i<j}\Pr_{hash\leftarrow\mathcal{H}}[hash(q_i) = 0]\right) \\
&\quad \overset{(0.0.3)}{\geq} \frac{1}{4(h+v)}.
\end{aligned} \tag{0.0.4}$$

We denote the set of those (π, ρ) for which $CanonicalSuccess^{P,C,hash}(\pi, \rho) = 1$ by $\mathcal{P}_{Canonical}$. For (π, ρ) for which C succeeds canonically, we have $Success^{P,C}(\pi, \rho) = 1$. Hence, $\mathcal{P}_{Canonical} \subseteq \mathcal{P}_{Success}$, and we conclude

$$\begin{aligned}
&\Pr_{\substack{hash\leftarrow\mathcal{H}\\ \pi,\rho}}\left[CanonicalSuccess^{P,C,hash}(\pi^{(k)},\rho) = 1\right] \\
&\quad = \Pr_{\substack{hash\leftarrow\mathcal{H}\\ (\pi,\rho)\in\mathcal{P}_{Success}}}[hash(q_j) = 0 \wedge (\forall i < j : hash(q_i) \neq 0)] \\
&\quad = \mathop{\mathbb{E}}_{(\pi,\rho)\in\mathcal{P}_{Success}}\left[\Pr_{hash\leftarrow\mathcal{H}}[hash(q_j) = 0 \wedge (\forall i < j : hash(q_i) \neq 0)]\right] \\
&\quad \overset{(0.0.4)}{\geq} \frac{\gamma}{4(h+v)}.
\end{aligned} \tag{0.0.5}$$

Algorithm: FindHash(γ, n, h, v)

Oracle: A problem poser P, a solver circuit C for P.
Input: Parameters γ, n, h, v
Output: A function $hash : Q \to \{0, 1, \ldots, 2(h+v) - 1\}$.

---

for $i = 1$ **to** $32(h+v)^2/\gamma^2$ **do:**
    $hash \leftarrow \mathcal{H}$
    $count := 0$
    for $j := 1$ **to** $32(h+v)^2/\gamma^2$ **do:**
        $\pi \overset{\$}{\leftarrow} \{0,1\}^n$
        $\rho \overset{\$}{\leftarrow} \{0,1\}^*$
        if $CanonicalSuccess^{P,C,hash}(\pi, \rho) = 1$ **then**
            $count := count + 1$
    if $\frac{\gamma^2}{32(h+v)^2} count \geq \frac{\gamma}{12(h+v)}$ **then**
        return $hash$
return $\perp$

We show that **FindHash** chooses $hash \in \mathcal{H}$ such that the canonical success probability of C with respect to $hash$ is at least $\frac{\gamma}{16(h+v)}$ almost surely. Let $\mathcal{H}_{Good}$ denote a family of functions $hash \in \mathcal{H}$ for which

$$\Pr_{\pi,\rho}\left[CanonicalSuccess^{P,C,hash}(\pi,\rho)=1\right] \geq \frac{\gamma}{8(h+v)}, \tag{0.0.6}$$

and $\mathcal{H}_{Bad}$ be the family of functions $hash \in \mathcal{H}$ such that

$$\Pr_{\pi,\rho}\left[CanonicalSuccess^{P,C,hash}(\pi,\rho)=1\right] \leq \frac{\gamma}{16(h+v)}. \tag{0.0.7}$$

Let N denote the number of iterations of the inner loop of **FindHash**. For a fixed $hash$, we define independent, identically distributed binary random variables $X_1, \ldots, X_N$ such that

$$X_i = \begin{cases} 1 & \text{if in the } i\text{-th iteration of the inner loop } count \text{ is increased} \\ 0 & \text{otherwise.} \end{cases}$$

We show now that **FindHash** is unlikely to return $hash \in \mathcal{H}_{Bad}$. For $hash \in \mathcal{H}_{Bad}$ by (0.0.7) we have $\mathbb{E}_{\pi,\rho}[X_i] \leq \frac{\gamma}{16(h+v)}$. Therefore, for any fixed $hash \in \mathcal{H}_{Bad}$ using the Chernoff bound we get [2]

$$\Pr_{\pi,\rho}\left[\frac{1}{N}\sum_{i=1}^{N} X_i \geq \frac{\gamma}{12(h+v)}\right] \leq \Pr_{\pi,\rho}\left[\frac{1}{N}\sum_{i=1}^{N} X_i \geq (1+\frac{1}{3})\mathbb{E}[X_i]\right] \leq e^{-\frac{\gamma}{16(h+v)}N/27} \leq e^{-\frac{2}{27}\frac{(h+v)}{\gamma}}.$$

The probability that $hash \in \mathcal{H}_{Good}$, when picked, is not returned amounts

$$\Pr_{\pi,\rho}\left[\frac{1}{N}\sum_{i=1}^{N} X_i \leq \frac{\gamma}{12(h+v)}\right] \leq \Pr_{\pi,\rho}\left[\frac{1}{N}\sum_{i=1}^{N} X_i \leq (1-\frac{1}{3})\mathbb{E}[X_i]\right] \leq e^{-\frac{\gamma}{8(h+v)}N/18} \leq e^{-\frac{2}{9}\frac{(h+v)}{\gamma}},$$

where we once more used the Chernoff bound. Now we show that the probability of picking a $hash \in \mathcal{H}_{Good}$ is at least $\frac{\gamma}{8(h+v)}$. We proof this statement by contradiction. We assume otherwise, namely that

$$\Pr_{hash \leftarrow \mathcal{H}}[hash \in \mathcal{H}_{Good}] < \frac{\gamma}{8(g+v)}. \tag{0.0.8}$$

We have

$$\begin{aligned}
&\Pr_{\substack{\pi,\rho \\ hash \leftarrow \mathcal{H}}}[CanonicalSuccess^{P,C,hash}(\pi,\rho)=1] \\
&\quad= \Pr_{\substack{\pi,\rho \\ hash \leftarrow \mathcal{H}}}[CanonicalSuccess^{P,C,hash}(\pi,\rho)=1 \mid hash \in \mathcal{H}_{Good}] \Pr_{hash \leftarrow \mathcal{H}}[hash \in \mathcal{H}_{Good}] \\
&\qquad+ \Pr_{\substack{\pi,\rho \\ hash \leftarrow \mathcal{H}}}[CanonicalSuccess^{P,C,hash}(\pi,\rho)=1 \mid hash \notin \mathcal{H}_{Good}] \Pr_{hash \leftarrow \mathcal{H}}[hash \notin \mathcal{H}_{Good}] \\
&\quad\leq \Pr_{hash \leftarrow \mathcal{H}}[hash \in \mathcal{H}_{Good}] + \Pr_{\substack{\pi,\rho \\ hash \leftarrow \mathcal{H}}}[CanonicalSuccess^{P,C,hash}(\pi,\rho)=1 \mid hash \notin \mathcal{H}_{Good}] \\
&\quad\overset{\substack{(0.0.6)\\(0.0.8)}}{<} \frac{\gamma}{8(h+v)} + \frac{\gamma}{8(h+v)} = \frac{\gamma}{4(h+v)},
\end{aligned}$$

but this contradicts (0.0.5). Therefore, we know that the probability of choosing a $hash \in \mathcal{H}_{Good}$ amounts at least $\frac{\gamma}{8(h+v)}$ where the probability is taken over a choice of $hash$.

[2] For $X = \sum_{i=1}^{N} X_i$ and $0 < \delta \leq 1$ we use the Chernoff bounds in the form $\Pr[X \geq (1+\delta)\mathbb{E}[X]] \leq e^{-\mathbb{E}[X]\delta^2/3}$ and $\Pr[X \leq (1-\delta)\mathbb{E}[X]] \leq e^{-\mathbb{E}[X]\delta^2/2}$.

We show that **FindHash** picks in one of its iteration $hash \in \mathcal{H}_{Good}$ almost surely. Let K be the number of iterations of the outer loop of **FindHash** and Y_i be a random variable for the event that in the i-th iteration of the outer loop $hash \notin \mathcal{H}_{Good}$ is picked. We conclude using $\Pr_{hash \leftarrow \mathcal{H}}[hash \notin \mathcal{H}_{Good}] < \frac{\gamma}{8(g+v)}$ and $K \leq \frac{32(h+v)^2}{\gamma^2}$ that

$$\Pr_{hash \leftarrow \mathcal{H}}[\bigcap_{1 \leq i \leq K} Y_i] \leq \left(1 - \frac{\gamma}{8(h+v)}\right)^K \leq e^{-\frac{\gamma}{8(h+v)}K} \leq e^{-\frac{4(h+v)}{\gamma}}. \qquad \square$$

Circuit $\widetilde{C}_2^{\Gamma_H^{(k)}, C_2, hash}(x, \rho)$

Oracle: A hint circuit $\Gamma_H^{(k)}$, a circuit C_2,
a function $hash : Q \to \{0, 1, \ldots, 2(h+v) - 1\}$.
Input: Bitstrings $x \in \{0,1\}^*$, $\rho \in \{0,1\}^*$.
Output: A tuple $(q, y_1, \ldots, y_k)$ or $\perp$.

Run $C_2^{\Gamma_V^{(g)}, \Gamma_H^{(k)}}(x, \rho)$
    if C_2 asks a hint query on q **then**
        if $hash(q) = 0$ **then**
            return $\perp$
        else
            answer the query of C_2 using $\Gamma_H^{(k)}(q)$

    if C_2 asks a verification query $(q, y_1, \ldots, y_k)$ **then**
        if $hash(q) = 0$ **then**
            return $(q, y_1, \ldots, y_k)$
        else
            answer the verification query with 0
return $\perp$

Given $C = (C_1, C_2)$ we define a circuit $\widetilde{C} = (C_1, \widetilde{C}_2)$. Every hint query q asked by $\widetilde{C}$ is such that $hash(q) \neq 0$. Furthermore, $\widetilde{C}$ asks no verification queries, instead it returns $\perp$ or (q, y) such that $hash(q) = 0$.

We say that for a fixed π, ρ the circuit $\widetilde{C}$ succeeds if for $x := \langle P(\pi), C_1(\rho)\rangle_{\text{trans}}$, $(\Gamma_V^{(g)}, \Gamma_H^{(k)}) := \langle P(\pi), C_1(\rho)\rangle_P$ we have

$$\Gamma_V^{(g)}(\widetilde{C}_2^{\Gamma_H^{(k)}, C_2, hash}(x, \rho)) = 1.$$

Lemma 1.5 *For fixed P, C and hash the following statement is true*

$$\Pr_{\pi,\rho}[CanonicalSuccess^{P,C,hash}(\pi, \rho) = 1] \leq \Pr_{\substack{\pi,\rho \\ x := \langle P(\pi), C_1(\rho)\rangle_{trans} \\ (\Gamma_V^{(g)}, \Gamma_H^{(k)}) := \langle P(\pi), C_1(\rho)\rangle_P}}[\Gamma_V^{(g)}(\widetilde{C}_2^{\Gamma_H^{(k)}, C_2, hash}(x, \rho)) = 1]$$

Proof. If for some fixed π, ρ and $hash$ the circuit C succeeds canonically, then for the same π, ρ and $hash$ also $\widetilde{C}$ succeeds. Using this observation, we conclude that

$$\begin{aligned}
&\Pr_{\pi,\rho}\left[CanonicalSuccess^{P,C,hash}(\pi,\rho)=1\right] \\
&\quad= \mathop{\mathbb{E}}_{\pi,\rho}\left[CanonicalSuccess^{P,C,hash}(\pi,\rho)=1\right] \\
&\quad\le \mathop{\mathbb{E}}_{\substack{\pi,\rho\\ x:=\langle P(\pi),C_1(\rho)\rangle_{\mathrm{trans}}\\ (\Gamma_V^{(g)},\Gamma_H^{(k)}):=\langle P(\pi),C_1(\rho)\rangle_P}}[\Gamma_V^{(g)}(C_2^{\Gamma_H^{(k)},C_2,hash}(x,\rho))=1] \\
&\quad\le \Pr_{\substack{\pi,\rho\\ x:=\langle P(\pi),C_1(\rho)\rangle_{\mathrm{trans}}\\ (\Gamma_V^{(g)},\Gamma_H^{(k)}):=\langle P(\pi),C_1(\rho)\rangle_P}}[\Gamma_V^{(g)}(\widetilde{C}_2^{\Gamma_H^{(k)},C_2,hash}(x,\rho))=1]
\end{aligned}$$

$\square$

Lemma 1.6 ***(Security amplification of a dynamic weakly verifiable puzzle with respect to hash.)*** *For fixed $P^{(1)}$ there exists an algorithm Gen that takes as input parameters $\varepsilon, \delta, n, k$, and outputs a circuit $D := (D_1, D_2)$ such that the following holds:*
If $\widetilde{C} := (C_1, \widetilde{C}_2)$ with oracle access to a solver circuit $C := (C_1, C_2)$ for $P^{(g)}$ is such that

$$\Pr_{\substack{\pi^{(k)},\rho\\ x:=\langle P^{(g)}(\pi^{(k)}),C_1(\rho)\rangle_{trans}\\ (\Gamma_H^{(k)},\Gamma_V^{(g)}):=\langle P^{(g)}(\pi^{(k)}),C_1(\rho)\rangle_{P^{(g)}}}}[\Gamma_V^{(g)}(\widetilde{C}_2^{\Gamma_H^{(k)},C_2,hash}(x,\rho))=1] \ge \Pr_{u\leftarrow\mu_\delta^k}[g(u)=1]+\varepsilon,$$

then D satisfies almost surely

$$\Pr_{\substack{\pi,\rho\\ x:=\langle P^{(1)}(\pi),D_1(\rho)\rangle_{trans}\\ (\Gamma_H,\Gamma_V):=\langle P^{(1)}(\pi),D_1(\rho)\rangle_{P^{(1)}}}}[\Gamma_V(D_2(x,\rho))=1] \ge (\delta+\frac{\varepsilon}{6k}).$$

Additionally, D and Gen require oracle access to g, $P^{(1)}$, $\widetilde{C}$. Furthermore, D asks at most $\frac{6k}{\varepsilon}\log\left(\frac{6k}{\varepsilon}\right) h$ hint queries and no verification queries. Finally, $Size(D) \le Size(C)\frac{6k}{\varepsilon}$ and $Time(Gen) = poly(k, \frac{1}{\varepsilon}, n, v, h)$.

Before proving Lemma 1.6 we define additional algorithms that are later used by Gen. First, we are interested in probability that for $u \leftarrow \mu_\delta^k$ and a bit $b \in \{0,1\}$ a function g with the first input bit set to b takes value 1. The estimate of this probability is calculated by the algorithm $EstimateFunctionProbability$.

EstimateFunctionProbability$^g(b, k, \varepsilon, \delta)$

Oracle: A function $g : \{0,1\}^k \to \{0,1\}$.
Input: A bit $b \in \{0,1\}$, parameters k, ε, δ.
Output: An estimate of $\Pr_{u\leftarrow\mu_\delta^k}[g(b,u_2,\ldots,u_k)=1]$.

---

for $i := 1$ **to** $\frac{16k^2}{\varepsilon^2}n$ **do:**
  $u \leftarrow \mu_\delta^{(k)}$
  $g_i := g(b, u_2, \ldots, u_k)$
return $\frac{\varepsilon^2}{16k^2n}\sum_{i=1}^{\frac{16k^2}{\varepsilon^2}n} g_i$

Lemma 1.7 *The procedure* ***EstimateFunctionProbability***$^g(b,k,\varepsilon,\delta)$ *outputs an estimate* $\widetilde{g}$ *of* $\Pr_{u\leftarrow\mu_\delta^k}[g(b,u_2,\ldots,u_k)=1]$ *where* $b\in\{0,1\}$ *such that* $|\widetilde{g}-\Pr_{u\leftarrow\mu_\delta^k}[g(b,u_2,\ldots,u_k)=1]|\leq\frac{\varepsilon}{8k}$ *almost surely.*

Proof. We define independent, identically distributed binary random variables $K_1,K_2,\ldots,K_{\frac{16k^2}{\varepsilon^2}n}$ such that for each $1\leq i\leq\frac{16k^2}{\varepsilon^2}n$ the random variable K_i equals g_i. We use the Chernoff bound to obtain [3]

$$\Pr\left[\left|\left(\frac{\varepsilon^2}{16k^2n}\sum_{i=1}^{\frac{16k^2}{\varepsilon^2}n}K_i\right)-\mathbb{E}[K_i]\right|\geq\frac{\varepsilon}{8k}\right]\leq 2\cdot e^{-n/3}. \qquad\square$$

The next algorithm **EvalutePuzzles**$^{P^{(1)},P^{(g)},\widetilde{C},hash}(\pi^{(k)},\rho)$ evaluates which puzzles defined by $P^{(g)}(\rho)$ are solved successfully by $\widetilde{C}(\rho)$. In this algorithm we use circuit $P^{(1)}$ invoking it k times to simulate k-rounds of interaction with $C_1(\rho)$. This requires additional notation. We write $\langle P^{(1)}(\pi_i),C_1(\rho)\rangle^i$ to denote the execution of the i-th round of the simulation, and $\langle P^{(1)}(\pi_i),C_1(\rho)\rangle^i_{P^{(1)}}$ to denote the output of $P^{(1)}(\pi_i)$ in the i-th round.

EvaluatePuzzles$^{P^{(1)},P^{(g)},\widetilde{C},hash}(\pi^{(k)},\rho)$

Oracle: Problem posers $P^{(1)}$, $P^{(g)}$, a circuit $\widetilde{C}=(C_1,\widetilde{C}_2)$,
a function $hash:Q\to\{0,1,\ldots,2(h+v)-1\}$.
Input: Bitstrings $\pi^{(k)}\in\{0,1\}^{kn}$, $\rho\in\{0,1\}^*$.
Output: A tuple $(c_1,\ldots,c_k)\in\{0,1\}^k$.

Run $\langle P^{(g)}(\pi^{(k)}),C_1(\rho)\rangle$
  $(\Gamma_V^{(g)},\Gamma_H^{(k)}):=\langle P(\pi^{(k)}),C_1(\rho)\rangle_{P^{(g)}}$
  $x:=\langle P^{(g)}(\pi^{(k)}),C_1(\rho)\rangle_{\text{trans}}$
$(q,y_1,\ldots,y_k):=\widetilde{C}_2^{\Gamma_H^{(k)},C_2,hash}(x,\rho)$
for $i:=1$ **to** k **do:** //simulate k rounds of interaction
  $(\Gamma_V^i,\Gamma_H^i):=\langle P^{(1)}(\pi_i),C_1(\rho)\rangle^i_{P^{(1)}}$
$(c_1,\ldots,c_k):=(\Gamma_V^1(q,y_1),\ldots,\Gamma_V^k(q,y_k))$
return $(c_1,\ldots,c_k)$

All puzzles used by the procedure **EvalutePuzzles** are generated internally and the algorithm can answer itself all queries to hint and verification oracle.

For fixed $\pi^{(k)},\rho$ let $(\Gamma_V^{(g)},\Gamma_H^{(k)}):=\langle P^{(g)}(\pi^{(k)}),C_1(\rho)\rangle_{P^{(g)}}$ and $x:=\langle P^{(g)}(\pi^{(k)}),C_1(\rho)\rangle_{\text{trans}}$. Additionally, we denote by (Γ_V^i,Γ_H^i) the verification and hint circuits generated in the i-th round of the interaction between $P^{(g)}(\pi^{(k)})$ and $C_1(\rho)$. Finally, for $(q,y_1,\ldots,y_k):=\widetilde{C}_2(x^{(k)},\rho)$ we denote the output of $\Gamma_V^i(q,y_i)$ by c_i.

We are interested in the success probability of $\widetilde{C}$ when the bitstring π_1 is fixed to π^*, the fact whether $\widetilde{C}$ succeeds in solving the first puzzle defined by $P^{(1)}(\pi_1)$ is neglected, and instead the bit $b\in\{0,1\}$ is used as the input on the first position to g. More formally, we define the surplus as

$$S_{\pi^*,b}=\Pr_{\pi^{(k)},\rho}[g(b,c_2,\ldots,c_k)=1\mid\pi_1=\pi^*]-\Pr_{u\leftarrow\mu^{(k)}}[g(b,u_2,\ldots,u_k)=1] \qquad (0.0.9)$$

[3]For independent Bernoulli distributed random variables $X_1,\ldots,X_n$ with $X:=\sum_{i=1}^n X_i$ and $0\leq\delta\leq1$ we use the Chernoff bound in the form $\Pr[|X-\mathbb{E}[X]|\geq\delta\mathbb{E}[X]]\leq 2e^{-\mathbb{E}[X]\delta^2/3}$.

The algorithm **EstimateSurplus** returns an estimate $\widetilde{S}_{\pi^*,b}$ for $S_{\pi^*,b}$.

EstimateSurplus$^{P^{(1)},P^{(g)},\widetilde{C},g,hash}(\pi^*, b, k, \varepsilon, \delta)$

Oracle: Posers $P^{(1)}$, $P^{(g)}$, a circuit $\widetilde{C}$, a function $g : \{0,1\}^k \to \{0,1\}$
a function $hash : Q \to \{0, 1, \ldots, 2(h+v)-1\}$.
Input: A bistring $\pi^* \in \{0,1\}^n$, a bit $b \in \{0,1\}$, parameters k, ε, δ.
Output: An estimate $\widetilde{S}_{\pi^*,b}$ for $S_{\pi^*,b}$.

$\widetilde{g}_b :=$ **EstimateFunctionProbability**$^g(b, k, \varepsilon, \delta)$
for $i := 1$ **to** $\frac{16k^2}{\varepsilon^2}n$ **do:**
  $(\pi_2, \ldots, \pi_k) \xleftarrow{\$} \{0,1\}^{(k-1)n}$
  $\rho \xleftarrow{\$} \{0,1\}^*$
  $(c_1, \ldots, c_k) :=$ **EvalutePuzzles**$^{P^{(1)},P^{(g)},\widetilde{C},hash}(\pi^*, \pi_2, \ldots, \pi_k, \rho)$
  $\widetilde{s}^i_{\pi^*,b} := g(b, c_2, \ldots, c_k)$
return $\left(\frac{\varepsilon^2 n}{16k^2} \sum_{i=1}^{\frac{16k^2}{\varepsilon^2}n} \widetilde{s}^i_{\pi^*,b}\right) - \widetilde{g}_b$

Lemma 1.8 *The estimate $\widetilde{S}_{\pi^*,b}$ returned by **EstimateSurplus** differs from $S_{\pi^*,b}$ by at most $\frac{\varepsilon}{4k}$ almost surely.*

Proof. We use the union bound and similar argument as in Lemma 1.7 which yields that $\frac{\varepsilon^2}{16k^2 n} \sum_{i=1}^{\frac{16k^2}{\varepsilon^2}n} \widetilde{s}^i_{\pi^*,b}$ differs from $\mathbb{E}[g(b, c_2, \ldots, c_k)]$ by at most $\frac{\varepsilon}{8k}$ almost surely. Together, with Lemma 1.7 we conclude that the surplus estimate returned by **EstimateSurplus** differs from $S_{\pi^*,b}$ by at most $\frac{\varepsilon}{4k}$ almost surely. □

We are ready to define the circuit D and the algorithm Gen.

Circuit $D = (D_1, D_2)(\rho)$

Phase I $D_1^{P^{(1)},\widetilde{C}}(\rho)$

Oracle: A circuit $\widetilde{C} = (C_1, \widetilde{C}_2)$, a poser $P^{(1)}$.
Input: A bitstring $\rho \in \{0,1\}^*$.

Interact with the problem poser $P^{(1)}$ using $C_1(\rho)$.
  Let x^* be the transcript of any internal simulations of C_1 and the interaction with the problem poser $P^{(1)}$.
  Let Γ_V^*, Γ_H^* be the verification and hint circuits output by the problem poser $P^{(1)}$.

Phase II $D_2^{P^{(1)},C,hash,g,\Gamma_V^*,\Gamma_H^*}(x^*, \rho)$

Oracle: A poser $P^{(1)}$, a solver circuit $\widetilde{C} = (C_1, \widetilde{C}_2)$,
functions $hash : Q \to \{0, 1, \ldots, 2(h+v)-1\}$, $g : \{0,1\}^k \to \{0,1\}$,
verification and hint circuits Γ_V^*, Γ_H^*.
Input: Bitstrings $x^* \in \{0,1\}^*$, $\rho \in \{0,1\}^*$.
Output: A verification query (q, y^*).

---

for at most $\frac{6k}{\varepsilon}\log(\frac{6k}{\varepsilon})$ iterations **do:**
  $\pi^{(k-1)} \leftarrow$ read $(k-1)\cdot n$ bits from ρ
  for $i := 2$ **to** k **do:** // Finish remaining $k-1$ interactions.
    Simulate $\langle P^{(1)}(\pi_i), C_1(\rho)\rangle$
      $x_i := \langle P^{(1)}(\pi_i), C_1(\rho)\rangle_{\text{trans}}$
      $(\Gamma_V^i, \Gamma_H^i) := \langle P^{(1)}(\pi_i), C_1(\rho)\rangle_{P^{(1)}}$
  $\Gamma_V^{(g)} := g(\Gamma_V^*, \Gamma_V^2, \ldots, \Gamma_V^k)$
  $\Gamma_H^{(k)} := (\Gamma_H^*, \Gamma_H^2, \ldots, \Gamma_H^k)$
  $(q, y^*, y_2, \ldots, y_k) := \widetilde{C}^{\Gamma_V^{(g)}, \Gamma_H^{(k)}, C, hash}((x^*, x_2, \ldots, x_k), \rho)$
  $(c^*, c_2, \ldots, c_k) := (\Gamma_V^*(q, y^*), \Gamma_V^2(q, y_2), \ldots, \Gamma_V^k(q, y_k))$
  if $g(1, c_2, \ldots, c_k) = 1 \wedge g(0, c_2, \ldots, c_k) = 0$ **then**
    Make a verification query (q, y^*)
    return (q, y^*)
return $\perp$

Algorithm $Gen^{P^{(1)}, P^{(g)}, \widetilde{C}, g, hash}(\varepsilon, \delta, n, v, h, k)$

Oracle: Posers $P^{(1)}$, $P^{(g)}$, circuit $\widetilde{C}$, functions $g : \{0,1\}^k \to \{0,1\}$,
$hash : Q \to \{0, 1, \ldots, 2(h+v)-1\}$.
Input: Parameters ε, δ, n, k, the number of verification v and hint h queries.
Output: A circuit D.

for $i := 1$ **to** $\frac{6k}{\varepsilon}n$ **do:**
  $\pi^* \xleftarrow{\$} \{0,1\}^n$
  $\widetilde{S}_{\pi^*,0} :=$ **EstimateSurplus**$^{P^{(1)}, P^{(g)}, \widetilde{C}, g, hash}(\pi^*, 0)$
  $\widetilde{S}_{\pi^*,1} :=$ **EstimateSurplus**$^{P^{(1)}, P^{(g)}, \widetilde{C}, g, hash}(\pi^*, 1)$
  if $\exists b \in \{0,1\} : \widetilde{S}_{\pi^*,b} \geq (1 - \frac{3}{4k})\varepsilon$ **then**
    Let C_1' be as C_1 except the first round of interaction between C_1 and $P^{(g)}$ which is simulated internally by using $P^{(1)}(\pi^*)$
    Let C_2' be as C_2 except the solution for the first puzzle which is discarded.
    $C' := (C_1', C_2')$
    $g'(b_2, \ldots, b_k) := g(b, b_2, \ldots, b_k)$
    return $Gen^{C', P^{(1)}, g', hash}(\varepsilon, \delta, n, v, h, k-1)$
// all estimates are lower than $(1 - \frac{3}{4k})\varepsilon$
return $D^{P^{(1)}, \widetilde{C}}$

Proof (Lemma 1.6). First let us consider the case where $k = 1$. The function $g : \{0,1\} \to \{0,1\}$ is either the identity or a constant function. If g is the identity function, then the circuit $D^{P^{(1)}, \widetilde{C}}$ returned by Gen directly uses $\widetilde{C}$ to find a solution. From the assumptions of Lemma 1.6 we know that $\widetilde{C}$ succeeds with probability at least $\delta + \varepsilon$. Hence, $D^{\widetilde{C}}$ trivially satisfies the statement. When g is a constant function the statement is vacuously true.

The general case is more involved. We distinguish two possibilities. If Gen manages to find in one of the iterations π^* with an estimate that satisfies $\widetilde{S}_{\pi^*,b} \geq (1 - \frac{3}{4k})\varepsilon$, we define a monotone function $g'(b_2, \ldots, b_k) := g(b, b_2, \ldots, b_k)$, and a circuit $\widetilde{C}' = (C_1', C_2')$, where C_1' first internally simulates the interaction between C_1 and $P^{(1)}(\pi^*)$, and then interacts with $P^{(g')}$. The circuit C_2' is defined as C_2 with the solution for the first puzzle discarded. In this case the surplus estimate is greater or equal $1 - \frac{3}{4k}\varepsilon$, and using Lemma 1.8 we conclude that $S_{\pi^*,b} \geq \widetilde{S}_{\pi^*,b} - \frac{\varepsilon}{4k} \geq 1 - \frac{\varepsilon}{4k}$ almost surely. The circuit $\widetilde{C}$ succeeds in solving $(k-1)$-wise direct product of puzzles with probability at least $\Pr_{u\leftarrow\mu_\delta^{k-1}}[g'(u_1,\ldots,u_{k-1})]+\varepsilon$. We see that in this case $\widetilde{C}'$ satisfies the conditions of Lemma 1.6 for $k-1$ puzzles and we can recurse using g' and $\widetilde{C}'$.

If all estimates are less than $(1-\frac{3}{4k})\varepsilon$, then intuitively C does not perform much better on the remaining $k-1$ puzzles than an algorithm that solves each puzzle independently with probability δ. However, from the assumption we know that on all k puzzles the performance of $\widetilde{C}$ is higher. Therefore, it is likely that the first puzzle is correctly solved unusual often. It remains to prove that this intuition is indeed correct. Let $\mathcal{G}_b := \{b_1, b_2, \ldots, b_k : g(b, b_2, \ldots, b_k) = 1\}$. We observe

$$\Pr_{u\leftarrow\mu_\delta^k}[u \in G_b] = \Pr_{u\leftarrow\mu_\delta^k}[g(b,u_2,\ldots,u_k)=1]$$
$$\Pr_{\pi^{(k)},\rho}[c \in G_b] = \Pr_{\pi^{(k)},\rho}[g(b,c_2,\ldots,c_k)=1].$$

We fix π^* and use (0.0.9) to obtain

$$\Pr_{u\leftarrow\mu_\delta^k}[u\in G_1] - \Pr_{u\leftarrow\mu_\delta^k}[u\in G_0] = \Pr_{\pi^{(k)},\rho}[c\in G_1 \mid \pi_1=\pi^*] - \Pr_{\pi^{(k)},\rho}[c\in G_0 \mid \pi_1=\pi^*] - (S_{\pi^*,1}-S_{\pi^*,0}) \tag{0.0.10}$$

Since g is monotone we have that $\mathcal{G}_0 \subseteq \mathcal{G}_1$ and can write (0.0.10) as

$$\Pr_{u\leftarrow\mu_\delta^k}[u\in \mathcal{G}_1\setminus\mathcal{G}_0] = \Pr_{\pi^{(k)},\rho}[c\in\mathcal{G}_1\setminus\mathcal{G}_0 \mid \pi_1=\pi^*] - (S_{\pi^*,1}-S_{\pi^*,0}). \tag{0.0.11}$$

Still fixing $\pi_1=\pi^*$ we multiply both sides of (0.0.11) by

$$\Pr_{\substack{\rho\\ x:=\langle P^{(1)}(\pi^*),D^{\widetilde{C}}(\rho)\rangle_{\text{trans}}\\ (\Gamma_V,\Gamma_H):=\langle P^{(1)}(\pi^*),D^{\widetilde{C}}(\rho)\rangle_{P^{(1)}}}}[\Gamma_V(D^{\widetilde{C}}(x,\rho))=1] / \Pr_{u\leftarrow\mu_\delta^k}[u\in\mathcal{G}_1\setminus\mathcal{G}_0].$$

which yields

$$\begin{aligned}
&\Pr_{\substack{\rho\\ x:=\langle P^{(1)}(\pi^*),D^{\widetilde{C}}(\rho)\rangle_{\text{trans}}\\ (\Gamma_V,\Gamma_H):=\langle P^{(1)}(\pi^*),D^{\widetilde{C}}(\rho)\rangle_{P^{(1)}}}}[\Gamma_V(D^{\widetilde{C}}(x,\rho))=1]\\
&\quad= \Pr_{\substack{\rho\\ x:=\langle P^{(1)}(\pi^*),D^{\widetilde{C}}(\rho)\rangle_{\text{trans}}\\ (\Gamma_V,\Gamma_H):=\langle P^{(1)}(\pi^*),D^{\widetilde{C}}(\rho)\rangle_{P^{(1)}}}}[\Gamma_V(D^{\widetilde{C}}(x,\rho))=1] \Pr_{\pi^{(k)},\rho}[c\in\mathcal{G}_1\setminus\mathcal{G}_0 \mid \pi=\pi^*]\frac{1}{\Pr_{u\leftarrow\mu_\delta^k}[u\in\mathcal{G}_1\setminus\mathcal{G}_0]}\\
&\quad\quad- \Pr_{\substack{\rho\\ x:=\langle P^{(1)}(\pi^*),D^{\widetilde{C}}(\rho)\rangle_{\text{trans}}\\ (\Gamma_V,\Gamma_H):=\langle P^{(1)}(\pi^*),D^{\widetilde{C}}(\rho)\rangle_{P^{(1)}}}}[\Gamma_V(D^{\widetilde{C}}(x,\rho))=1](S_{\pi^*,1}-S_{\pi^*,0})\frac{1}{\Pr_{u\leftarrow\mu_\delta^k}[u\in\mathcal{G}_1\setminus\mathcal{G}_0]}
\end{aligned} \tag{0.0.12}$$

We make use of the fact that the event $c\in\mathcal{G}_1\setminus\mathcal{G}_0$ implies $D_2(x^*,r)\neq\perp$, and write the first summand of (0.0.12) as

$$\begin{aligned}
&\Pr_{\substack{\rho\\ x:=\langle P^{(1)}(\pi^*),D_1(\rho)\rangle_{\text{trans}}\\ (\Gamma_V,\Gamma_H):=\langle P^{(1)}(\pi^*),D_1(\rho)\rangle_{P^{(1)}}}}[\Gamma_V(D_2(x,\rho))=1] \Pr_{\pi^{(k)},\rho}[c\in\mathcal{G}_1\setminus\mathcal{G}_0 \mid \pi_1=\pi^*]\\
&\quad= \Pr_{\substack{\rho\\ x=\langle P^{(1)}(\pi^*),D_1(\rho)\rangle_{\text{trans}}}}[D_2(x,\rho)\neq\perp] \Pr_{\pi^{(k)},\rho}[c_1=1 \mid c\in\mathcal{G}_1\setminus\mathcal{G}_0, \pi_1=\pi^*] \Pr_{\pi^{(k)},\rho}[c\in\mathcal{G}_1\setminus\mathcal{G}_0 \mid \pi_1=\pi^*]
\end{aligned} \tag{0.0.13}$$

Now we consider two cases: if $\Pr_{\pi^{(k)},\rho}[c \in \mathcal{G}_1 \setminus \mathcal{G}_0 \mid \pi_1 = \pi^*] \le \frac{\varepsilon}{6k}$ then

$$\Pr_{\pi^{(k)}}[c_1 = 1 \mid c \in \mathcal{G}_1 \setminus \mathcal{G}_0, \pi_1 = \pi^*] \Pr_{\pi^{(k)}}[c \in \mathcal{G}_1 \setminus \mathcal{G}_0 \mid \pi_1 = \pi^*] \le \frac{\varepsilon}{6k}, \tag{0.0.14}$$

for $\Pr_{\pi^{(k)},\rho}[c \in \mathcal{G}_1 \setminus \mathcal{G}_0] > \frac{\varepsilon}{6k}$ the circuit D_2 outputs $\perp$ if and only if it fails in all $\frac{6k}{\varepsilon}\log(\frac{6k}{\varepsilon})$ iterations to find $\pi^{(k)}$ such that $g(1, c_2, \ldots, c_k) = 1 \wedge g(0, c_2, \ldots, c_k) = 0$ (i.e. in none of the iterations $c \in \mathcal{G}_1 \setminus \mathcal{G}_0$) which happens with probability

$$\Pr_{\substack{\rho \\ x^* := \langle P^{(1)}(\pi^*), D_1(\rho)\rangle_{\text{trans}}}}[D_2(x^*, \rho) = \perp] \le (1 - \frac{\varepsilon}{6k})^{\frac{6k}{\varepsilon}\log(\frac{\varepsilon}{6k})} \le \frac{\varepsilon}{6k}. \tag{0.0.15}$$

We conclude that in both cases:

$$\begin{aligned}
&\Pr_{\substack{\rho \\ x := \langle P^{(1)}(\pi^*), D_1(\rho)\rangle_{\text{trans}}}}[D_2(x, \rho) \ne \perp] \Pr_{\pi^{(k)},\rho}[c_1 = 1 \mid c \in \mathcal{G}_1 \setminus \mathcal{G}_0, \pi_1 = \pi^*] \Pr_{\pi^{(k)},\rho}[c \in \mathcal{G}_1 \setminus \mathcal{G}_0 \mid \pi_1 = \pi^*] \\
&\qquad \ge \Pr_{\pi^{(k)},\rho}[c_1 = 1 \mid c \in \mathcal{G}_1 \setminus \mathcal{G}_0, \pi_1 = \pi^*] \Pr_{\pi^{(k)},\rho}[c \in \mathcal{G}_1 \setminus \mathcal{G}_0 \mid \pi_1 = \pi^*] - \frac{\varepsilon}{6k}.
\end{aligned} \tag{0.0.16}$$

Therefore, we have

$$\begin{aligned}
&\Pr_{\substack{\rho \\ x^* := \langle P^{(1)}(\pi^*), D_1(\rho)\rangle_{\text{trans}}}}[D_2(x^*, \rho) \ne \perp] \Pr_{\pi^{(k)},\rho}[c_1 = 1 \mid c \in \mathcal{G}_1 \setminus \mathcal{G}_0, \pi_1 = \pi^*] \Pr_{\pi^{(k)},\rho}[c \in \mathcal{G}_1 \setminus \mathcal{G}_0 \mid \pi_1 = \pi^*] \\
&\qquad \ge \Pr_{\pi^{(k)},\rho}[c_1 = 1 \wedge c \in \mathcal{G}_0 \setminus \mathcal{G}_1 \mid \pi_1 = \pi^*] - \frac{\varepsilon}{6k} \\
&\qquad = \Pr_{\pi^{(k)},\rho}[g(c_1, c_2, \ldots, c_k) = 1 \wedge g(0, c_2, \ldots, c_k) = 0 \mid \pi_1 = \pi^*] - \frac{\varepsilon}{6k} \\
&\qquad = \Pr_{\pi^{(k)},\rho}[g(c_1, c_2, \ldots, c_k) = 1 \mid \pi_1 = \pi^*] - \Pr_{\pi^{(k)},\rho}[c \in \mathcal{G}_0 \mid \pi_1 = \pi^*] - \frac{\varepsilon}{6k},
\end{aligned}$$

and finally by (0.0.9)

$$\begin{aligned}
&\Pr_{\substack{\rho \\ x^* := \langle P^{(1)}(\pi^*), D_1(\rho)\rangle_{\text{trans}}}}[D_2(x^*, \rho) \ne \perp] \Pr_{\pi^{(k)},\rho}[c_1 = 1 \mid c \in \mathcal{G}_1 \setminus \mathcal{G}_0, \pi_1 = \pi^*] \Pr_{\pi^{(k)},\rho}[c \in \mathcal{G}_1 \setminus \mathcal{G}_0 \mid \pi_1 = \pi^*] \\
&\qquad = \Pr_{\pi^{(k)},\rho}[g(c_1, c_2, \ldots, c_k) = 1 \mid \pi_1 = \pi^*] - \Pr_{u \leftarrow \mu_\delta^{(k)}}[u \in \mathcal{G}_0] - S_{\pi^*,0} - \frac{\varepsilon}{6k}.
\end{aligned} \tag{0.0.17}$$

For the second summand we show that if we do not recurse, then majority of estimates is low almost surely. Let assume

$$\Pr_{\pi,\rho}\left[\left(S_{\pi,0} \le (1 - \frac{1}{2k})\varepsilon\right) \wedge \left(S_{\pi,1} \le (1 - \frac{1}{2k})\varepsilon\right)\right] < 1 - \frac{\varepsilon}{6k}, \tag{0.0.18}$$

then the algorithm recurses almost surely. Therefore, under the assumption that *Gen* does not recurse, we have almost surely

$$\Pr_{\pi,\rho}\left[\left(S_{\pi,0} \le (1 - \frac{1}{2k})\varepsilon\right) \wedge \left(S_{\pi,1} \le (1 - \frac{1}{2k})\varepsilon\right)\right] \ge 1 - \frac{\varepsilon}{6k}. \tag{0.0.19}$$

Let us define a set

$$\mathcal{W} = \left\{\pi : \left(S_{\pi,0} \le (1 - \frac{1}{2k})\varepsilon\right) \wedge \left(S_{\pi,1} \le (1 - \frac{1}{2k})\varepsilon\right)\right\} \tag{0.0.20}$$

and use $\mathcal{W}^c$ to denote the complement of $\mathcal{W}$. We bound the second summand in (0.0.12)

$$\begin{aligned}
&\mathbb{E}_{\pi^*}[S_{\pi^*,0} + \Pr_{\substack{\rho \\ x:=\langle P^{(1)}(\pi^*),D_1(\rho)\rangle_{\text{trans}} \\ (\Gamma_V,\Gamma_H):=\langle P^{(1)}(\pi^*),D_1(\rho)\rangle_{P(1)}}}[\Gamma_V(D_2^{\widetilde{C}}(x,\rho)) = 1](S_{\pi^*,1} - S_{\pi^*,0})] \\
&\quad = \mathbb{E}_{\pi^*\in\mathcal{W}^c}[S_{\pi^*,0} + \Pr_{\substack{\rho \\ x:=\langle P^{(1)}(\pi^*),D_1(\rho)\rangle_{\text{trans}} \\ (\Gamma_V,\Gamma_H):=\langle P^{(1)}(\pi^*),D_1(\rho)\rangle_{P(1)}}}[\Gamma_V(D_2^{\widetilde{C}}(x,\rho)) = 1](S_{\pi^*,1} - S_{\pi^*,0})] \\
&\qquad + \mathbb{E}_{\pi^*\in\mathcal{W}}[S_{\pi^*,0} + \Pr_{\substack{\rho \\ x:=\langle P^{(1)}(\pi^*),D_1(\rho)\rangle_{\text{trans}} \\ (\Gamma_V,\Gamma_H):=\langle P^{(1)}(\pi^*),D_1(\rho)\rangle_{P(1)}}}[\Gamma_V(D_2^{\widetilde{C}}(x,\rho)) = 1](S_{\pi^*,1} - S_{\pi^*,0})] \\
&\quad \le \frac{\varepsilon}{6k} + \mathbb{E}_{\pi^*\in\mathcal{W}^c}[S_{\pi^*,0} + \Pr_{\substack{\rho \\ x:=\langle P^{(1)}(\pi^*),D_1(\rho)\rangle_{\text{trans}} \\ (\Gamma_V,\Gamma_H):=\langle P^{(1)}(\pi^*),D_1(\rho)\rangle_{P(1)}}}[\Gamma_V(D_2^{\widetilde{C}}(x,\rho)) = 1]((1 - \frac{1}{2k})\varepsilon - S_{\pi^*,0})] \\
&\quad \le \frac{\varepsilon}{6k} + 1 - \frac{\varepsilon}{2k} = 1 - \frac{\varepsilon}{3k} \qquad (0.0.21)
\end{aligned}$$

We observe that

$$\Pr_{u\leftarrow\mu_\delta^k}[g(u) = 1] = \Pr[u \in \mathcal{G}_0 \vee (u \in \mathcal{G}_1 \setminus \mathcal{G}_0 \wedge u_1 = 1)] = \Pr[u \in \mathcal{G}_0] + \Pr[u \in \mathcal{G}_1 \setminus \mathcal{G}_0] \Pr[u_1 = 1]. \qquad (0.0.22)$$

Finally, we insert (0.0.17) and (0.0.21) into equation (0.0.12) and use (0.0.22) to obtain

$$\Pr_{\substack{\rho \\ x:=\langle P^{(1)}(\pi^*),D_1(\rho)\rangle_{\text{trans}} \\ (\Gamma_V,\Gamma_H):=\langle P^{(1)}(\pi^*),D_1(\rho)\rangle_{P(1)}}}[\Gamma_V(D_2^{\widetilde{C}}(x,\rho)) = 1] \ge \mathbb{E}_{\pi^*}\left[\frac{\Pr_{\pi^{(k)}}[g(c) = 1 \mid \pi_1 = \pi^*] - \Pr_{u\leftarrow\mu_\delta^k}[u \in G_0] - (1 - \frac{1}{6k})\varepsilon}{\Pr_{u\leftarrow\mu_\delta^k}[u \in \mathcal{G}_1 \setminus \mathcal{G}_0]}\right]$$

Using the assumptions of Lemma 1.6, we get

$$\begin{aligned}
\Pr_{\substack{\rho \\ x:=\langle P^{(1)}(\pi^*),D_1(\rho)\rangle_{\text{trans}} \\ (\Gamma_V,\Gamma_H):=\langle P^{(1)}(\pi^*),D_1(\rho)\rangle_{P(1)}}}[\Gamma_V(D_2^{\widetilde{C}}(x,\rho)) = 1] &\ge \frac{\Pr_{u\leftarrow\mu_\delta^k}[g(u) = 1] + \varepsilon + \Pr_{u\leftarrow\mu_\delta^k}[u \in \mathcal{G}_0] - (1 - \frac{1}{6k})\varepsilon}{\Pr_{u\leftarrow\mu_\delta^k}[u \in \mathcal{G}_1 \setminus \mathcal{G}_0]} \\
&\ge \frac{\varepsilon + \delta \Pr_{u\leftarrow\mu_\delta^k}[u \in \mathcal{G}_1 \setminus \mathcal{G}_0] - (1 - \frac{1}{6k})\varepsilon}{\Pr_{u\leftarrow\mu_\delta^k}[u \in \mathcal{G}_1 \setminus \mathcal{G}_0]} \ge \delta + \frac{\varepsilon}{6k} \qquad (0.0.23)
\end{aligned}$$

□

Proof (Theorem 1.3). We show that Theorem 1.3 follows by Lemmas: 1.6, 1.4. First given a solver circuit C such that

$$\Pr_{\pi^{(k)},\rho}\left[Success^{P^{(g)},C}(\pi^{(k)},\rho) = 1\right] \ge 8(h+v)\left(\Pr_{u\leftarrow\mu_\delta^k}[g(u) = 1] + \varepsilon\right)$$

we apply Lemma 1.4 to find a function *hash* such that

$$\Pr_{\pi^{(k)},\rho}\left[CanonicalSuccess^{P^{(g)},C,hash}(\pi^{(k)},\rho) = 1\right] \ge \Pr_{u\leftarrow\mu_\delta^k}[g(u) = 1] + \varepsilon.$$

By Lemma (1.5) we know that it is possible to create a circuit $\widetilde{C}$ with oracle access to *hash* and C such that

$$\Pr_{\substack{\pi,\rho \\ x:=\langle P(\pi),C_1(\rho)\rangle_{\text{trans}} \\ (\Gamma_V^{(g)},\Gamma_H^{(k)}):=\langle P(\pi),C_1(\rho)\rangle_P}}[\Gamma_V^{(g)}(\widetilde{C}_2^{\Gamma_H^{(k)},C_2,hash}(x,\rho)) = 1] \ge \Pr_{u\leftarrow\mu_\delta^k}[g(u) = 1] + \varepsilon$$

Now, we apply Lemma 1.6 for the function *hash* and the circuit $\widetilde{C}$ and obtain a circuit D such that

$$\Pr_{\substack{\pi,\rho \\ x:=\langle P^{(1)}(\pi),D_1(\rho)\rangle_{\text{trans}} \\ (\Gamma_H,\Gamma_V):=\langle P^{(1)}(\pi),D_1(\rho)\rangle_{P^{(1)}}}} [\Gamma_V(D_2(x,\rho)) = 1] \geq (\delta + \frac{\varepsilon}{6k}). \tag{0.0.24}$$

Finally, we use the circuit $\widetilde{D}$ that first runs the circuit D, and make a verification query using (q, y) returned by D. From, we know that probability that this query is successful amounts at least $(\delta + \frac{\varepsilon}{6k})$. Therefore, we have

$$\Pr_{\pi,\rho}\left[Success^{P^{(1)},\widetilde{D}}(\pi,\rho) = 1\right] \geq (\delta + \frac{\varepsilon}{6k}). \qquad \square$$