

Definition 1.1 (Dynamic weakly verifiable puzzle.) A dynamic weakly verifiable puzzle (DWVP) is defined by a probabilistic algorithm P called a problem poser. We write $P(\pi)$ to denote the algorithm P with the randomness π . The algorithm P outputs circuits Γ_V , Γ_H and a puzzle $x \in \{0, 1\}^*$. The circuit Γ_V takes as input $q \in Q$, an answer $y \in \{0, 1\}^*$, and outputs 1 if y is a correct solution of x for q and 0 otherwise. The circuit Γ_H on input $q \in Q$ outputs a hint such that $\Gamma_V(q, \Gamma_H(q)) = 1$.

A problem solver S is a probabilistic algorithm that takes as input a puzzle x , and has oracle access to Γ_V and Γ_H . The randomness used by S is denoted by ρ . The execution of S with the input x and the randomness ρ is denoted by $S(x, \rho)$. The queries of S to Γ_V are called verification queries, and to Γ_H hint queries. The solver S can ask at most h hint queries, v verification queries, and successfully solves the puzzle if and only if it makes a verification query (q, y) such that $\Gamma_V(q, y) = 1$, when it has not previously asked for a hint query on q .

Definition 1.2 (k -wise direct-product of DWVPs.) Let $g : \{0, 1\}^k \rightarrow \{0, 1\}$ 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 $P^{(g)}$. Let $\pi^{(k)} := (\pi_1, \dots, \pi_k)$ be the randomness used by $P^{(g)}$, and $P^{(g)}(\pi^{(k)})$ denote the execution of $P^{(g)}$ with the randomness $\pi^{(k)}$. The poser $P^{(g)}$ outputs: a verification circuit

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

a hint circuit

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

and a puzzle $x^{(k)} := (x_1, \dots, x_k)$, where the i -th instance $(x_i, \Gamma_V^i, \Gamma_H^i) := P^{(1)}(\pi_i)$.

We consider the following random experiment in which a DWVP, defined by P , is solved by a random circuit C .

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

Oracle: A problem poser P , a solver circuit $C^{(\cdot, \cdot)}$.

Input: Bitstrings π, ρ .

Output: A bit $b \in \{0, 1\}$.

$(x, \Gamma_V, \Gamma_H) := P(\pi)$

Run $C^{\Gamma_V, \Gamma_H}(x, \rho)$

Let $Q_{\text{Solved}} := \{q : C^{\Gamma_V, \Gamma_H} \text{ asked a verification query } (q, y) \text{ and } \Gamma_V(q, y) = 1\}$

Let $Q_{\text{Hint}} := \{q : C^{\Gamma_V, \Gamma_H} \text{ asked a hint query on } q\}$

If $\exists q \in Q_{\text{Solved}} : q \notin Q_{\text{Hint}}$ **then**

return 1

else

return 0

The success probability of C in solving DWVP posed by P is

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

TODO: Do we have to fix P here? **TODO:** Do the circuit bound is well defined? **TODO:** What happens when $8(h + v) \left(\Pr_{\mu \leftarrow \mu_\delta^k} [g(\mu) = 1] + \varepsilon \right) \geq 1$ then the formula does not work

TODO: Define $\mu_\delta^{(k)}$

Theorem 1.3 Security amplification for a dynamic weakly verifiable puzzle.

Let $P^{(1)}$ be a fixed problem poser as in Definition 1.1. There exists a probabilistic algorithm $\text{Gen}(C, g, \varepsilon, \delta, n, v, h)$ which takes as input a solver circuit C for the k -wise direct product of $P^{(1)}$, a monotone function $g : \{0, 1\}^k \rightarrow \{0, 1\}$, parameters ε, δ, n , the number of verification queries v , and hint queries h asked by C , and outputs a circuit D such that the following holds: If C is such that

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

then D satisfies almost surely

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

Additionally, D and Gen require only oracle access to g and C . Furthermore, D asks at most h hint queries, v verification queries and $\text{Size}(D) \leq \text{Size}(C) \cdot \Theta(\frac{6k}{\varepsilon})$ and $\text{Time}(\text{Gen}) = \text{poly}(k, \frac{1}{\varepsilon}, n, v, h)$.

The Theorem 1.3 implies that if there is no good solver for $P^{(1)}$, then a good solver for $P^{(k)}$ does not exist.

Intuitively, the algorithm Gen tries to find $k - 1$ puzzles and a position for an input puzzle x , such that when C runs with the $k - 1$ puzzles and x placed on this position, then x is often solved correctly. To find a good position for x and these $k - 1$ puzzles Gen needs to run C several times. In these runs C asks a hint queries. Let us assume that after some trials and errors Gen finally finds a set of puzzles and a position for x such that x is often solved. However, it may still not constitute a valid solution, as an additional requirement is needed that this happens often for q on which a hint query was not asked before. To satisfy this requirement we split Q .

Let $\text{hash} : Q \rightarrow \{0, 1, \dots, 2(h + v) - 1\}$, then a set $P_{\text{hash}} \subseteq Q$, defined with respect to hash , is a preimage of 0 for function hash . The set P_{hash} contains q on which C is not allowed to ask hint queries. Therefore, if C makes a verification query on $q \in P_{\text{hash}}$ we know that no hint query is ever asked on this q . In the experiment E a circuit C succeeds if and only if it ask a verification query on $q \in P_{\text{hash}}$ and no hint query is asked on $q \in P_{\text{hash}}$.

Experiment $E^{P^{(g)}, C^{(\cdot, \cdot)}, \text{hash}}(\pi_1, \dots, \pi_k, r)$

Oracle: A problem poser $P^{(g)}$ for a k -wise direct product.

A solver circuit $C^{(\cdot, \cdot)}$ for a k -wise direct product.

A function $\text{hash} : Q \leftarrow \{0, \dots, 2(h + v) - 1\}$.

Input: Random bitstrings: $(\pi_1, \dots, \pi_k) \in \{0, 1\}^{kl}$, r .

$$(x^k, \Gamma_V^{(g)}, \Gamma_H^{(k)}) := P^{(g)}(\pi^k)$$

Run $C^{\Gamma_V^{(g)}, \Gamma_H^{(k)}}(x^k, r)$

Let $(q_j, y_j^{(k)})$ be the first successful verification query if $C^{\Gamma_V^{(g)}, \Gamma_H^{(k)}}$ succeeds or an arbitrary verification query when it fails.

If $(\forall i < j : q_i \notin P_{hash})$ and $q_j \in P_{hash}$ and $\Gamma_V^{(g)}(q_j, y_j^{(k)}) = 1$
return 1
else
return 0

For fixed $hash$ and $P^{(1)}$ a canonical success of C is a situation when $E^{P^{(g)}, C^{(\cdot, \cdot)}, hash}(\pi_1, \dots, \pi_k, r) = 1$. We show that if C often solves successfully a k -wise direct product of DWVP in the random experiment A , then it also often succeeds canonically.

Lemma 1.4 *Success probability in solving a k -wise direct product of DWVP with respect to a function hash.*

For fixed $P^{(g)}$ let C succeed in solving a k -wise direct product of DWVP produced by $P^{(g)}$ with probability γ , asking at most h hint and v verification queries. There exists a probabilistic algorithm, with oracle access to C , that runs in time $O((h+v)^4/\gamma^4)$ and with high probability outputs a function $hash : Q \rightarrow \{0, \dots, 2(h+v) - 1\}$ such that the canonical success probability of C with respect to P_{hash} is at least $\frac{\gamma}{8(h+v)}$.

Proof. Let \mathcal{H} be a family of pairwise independent hash functions $Q \rightarrow \{0, 1, \dots, 2(h+v) - 1\}$. For all $i \neq j \in \{1, \dots, (h+v)\}$ and $k, l \in \{0, 1, \dots, 2(h+v) - 1\}$ by pairwise independence property of \mathcal{H} , we have

$$\forall q_i, q_j \in Q : \Pr_{hash \leftarrow \mathcal{H}}[hash(q_i) = k \mid hash(q_j) = l] = \Pr_{hash \leftarrow \mathcal{H}}[hash(q_i) = k] = \frac{1}{2(h+v)}. \quad (0.0.2)$$

Let $P^{(g)}, C, (\pi_1, \dots, \pi_k)$ be fixed. We consider the experiment E and define a binary random variable X for the event that $hash(q_j) = 0$, and for every query q_i asked before $q_j : hash(q_i) \neq 0$. Conditioned on the event $hash(q_i) = 0$, we get

$$\begin{aligned} \Pr_{hash \leftarrow \mathcal{H}}[X = 1] &= \Pr_{hash \leftarrow \mathcal{H}}[hash(q_j) = 0 \wedge \forall i < j : hash(q_i) \neq 0] \\ &= \Pr_{hash \leftarrow \mathcal{H}}[\forall i < j : hash(q_i) \neq 0 \mid hash(q_j) = 0] \Pr_{hash \leftarrow \mathcal{H}}[hash(q_j) = 0]. \end{aligned}$$

Now we use (0.0.2) twice and obtain

$$\begin{aligned} \Pr_{hash \leftarrow \mathcal{H}}[X = 1] &= \frac{1}{2(h+v)} \left(1 - \Pr_{hash \leftarrow \mathcal{H}}[\exists i < j : hash(q_i) = 0 \mid hash(q_j) = 0] \right) \\ &= \frac{1}{2(h+v)} \left(1 - \Pr_{hash \leftarrow \mathcal{H}}[\exists i < j : hash(q_i) = 0] \right). \end{aligned}$$

Finally, we use union bound and $j \leq (h+v)$ to get

$$\Pr_{hash \leftarrow \mathcal{H}}[X = 1] \geq \frac{1}{2(h+v)} \left(1 - \sum_{i < j} \Pr_{hash \leftarrow \mathcal{H}}[hash(q_i) = 0] \right) \geq \frac{1}{4(h+v)}.$$

Let G_A (correspondingly G_E) denote the set of all (π_1, \dots, π_k) for which C succeeds in the random experiment A (E). For fixed (π_1, \dots, π_k) , if C succeeds canonically, then it also succeeds in the random experiment A . Hence, $G_E \subseteq G_A$ and we get

$$\Pr_{\substack{hash \leftarrow \mathcal{H} \\ (\pi_1, \dots, \pi_k)}}[E^{P^{(g)}, C^{(\cdot, \cdot)}, hash}(\pi_1, \dots, \pi_k) = 1] = \mathbb{E}_{(\pi_1, \dots, \pi_k) \in G_A} \left[\Pr_{hash \leftarrow \mathcal{H}}[X = 1] \right] \geq \frac{\gamma}{4(h+v)}. \quad (0.0.3)$$

Algorithm: FindHash**Oracle:** A solver circuit $C^{(\cdot, \cdot)}$ for a k -wise direct product of DWVP.**Input:** A set \mathcal{H} .

```

For  $i = 1$  to  $32(h+v)^2/\gamma^2$ 
   $hash \xleftarrow{\$} \mathcal{H}$ 
   $count := 0$ 
  For  $j := 1$  to  $32(h+v)^2/\gamma^2$ 
     $(\pi_1, \dots, \pi_k) \xleftarrow{\$} \{0, 1\}^{kl}$ 
    If  $E^{P(g), C^{(\cdot, \cdot)}, hash}(\pi_1, \dots, \pi_k) = 1$  then
       $count := count + 1$ 
  If  $\frac{\gamma^2}{32(h+v)^2} count \geq \frac{\gamma}{6(h+v)}$ 
    return  $hash$ 
return  $\perp$ 

```

We show that **FindHash** chooses $hash$ such that the canonical success probability of C with respect to P_{hash} is at least $\frac{\gamma}{4(h+v)}$ almost surely. Let \mathcal{H}_{Good} denote a family of functions $hash \in \mathcal{H}$ for which

$$\Pr_{(\pi_1, \dots, \pi_k)} [E^{P(g), C^{(\cdot, \cdot)}, hash}(\pi_1, \dots, \pi_k) = 1] \geq \frac{\gamma}{4(h+v)},$$

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

$$\Pr_{(\pi_1, \dots, \pi_k)} [E^{P(g), C^{(\cdot, \cdot)}, hash}(\pi_1, \dots, \pi_k) = 1] \leq \frac{\gamma}{8(h+v)}.$$

Additionally, for a fixed $hash$, we define binary random variables $X_1, \dots, X_i, \dots, X_N$ such that

$$X_i = \begin{cases} 1 & \text{if in } i\text{th iteration variable } count \text{ is increased} \\ 0 & \text{otherwise.} \end{cases}$$

We first show that it is unlikely that **FindHash** returns $hash \in \mathcal{H}_{Bad}$. For $hash \in \mathcal{H}_{Bad}$ we have $\mathbb{E}_{(\pi_1, \dots, \pi_k)} [X_i] < \frac{\gamma}{8(h+v)}$. Therefore, for any fixed $hash \in \mathcal{H}_{Bad}$ using the Chernoff bound we get

$$\Pr_{(\pi_1, \dots, \pi_k)} \left[\frac{1}{N} \sum_{i=1}^N X_i \geq \frac{\gamma}{6(h+v)} \right] \leq \Pr_{(\pi_1, \dots, \pi_k)} \left[\frac{1}{N} \sum_{i=1}^N X_i \geq (1 + \frac{1}{3}) \mathbb{E}[X_i] \right] \leq e^{-\frac{\gamma}{4(h+v)} N/27}.$$

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

$$\Pr_{(\pi_1, \dots, \pi_k)} \left[\frac{1}{N} \sum_{i=1}^N X_i \leq \frac{\gamma}{6(h+v)} \right] \leq \Pr_{(\pi_1, \dots, \pi_k)} \left[\frac{1}{N} \sum_{i=1}^N X_i \leq (1 - \frac{1}{3}) \mathbb{E}[X_i] \right] \leq e^{-\frac{\gamma}{4(h+v)} N/27}.$$

Finally, we show that **FindHash** picks in one of its iteration $hash \in \mathcal{H}_{Good}$ almost surely. Let Y_i be a binary random variable such that

$$Y_i = \begin{cases} 1 & \text{if in } i\text{th iteration } hash \in \mathcal{H}_{Good} \text{ is picked} \\ 0 & \text{otherwise.} \end{cases}$$

From equation (0.0.3) we know that $\Pr_{hash \leftarrow \mathcal{H}} [Y_i = 1] = \mathbb{E}[Y_i] \geq \frac{\gamma}{4(h+v)}$, almost surely. Thus, we get

$$\Pr_{hash \leftarrow \mathcal{H}} \left[\sum_{i=1}^K Y_i = 0 \right] \leq \left(1 - \frac{\gamma}{4(h+v)} \right)^K \leq e^{-\frac{\gamma}{4(h+v)} K}.$$

The bound stated in the Lemma 1.4 is achieved for $\delta = \frac{1}{3}$ and $K = N = 32(h+v)^2/\gamma^2$. \square

Lemma 1.5 *Security amplification of a dynamic weakly verifiable puzzle with respect to P_{hash} .*

For fixed $P^{(1)}$ there exists an algorithm $Gen(C, g, \varepsilon, \delta, n, v, h, hash)$, which takes as input a solver circuit C , a monotone function g , a function $hash : Q \rightarrow \{0, \dots, 2(h+v)-1\}$, parameters ε, δ, n , number of verification v , and hint h queries asked by C , and outputs a circuit D such that following holds:

If C is such that

$$\Pr_{(\pi_1, \dots, \pi_k)} [E^{P^{(g)}, C, Hash}(\pi_1, \dots, \pi_k) = 1] \geq \Pr_{\mu \leftarrow \mu_\delta^k} [g(\mu) = 1] + \varepsilon,$$

then D satisfies almost surely

$$\Pr_\pi [\Gamma_V^{(g)}(D^{P^{(1)}, \tilde{C}, hash}(\pi)) = 1] \geq (\delta + \frac{\varepsilon}{6k}).$$

Furthermore, $Size(D) \leq Size(C) \frac{6k}{\varepsilon}$ and $Time(Gen) = poly(k, \frac{1}{\varepsilon}, n, v, h)$.

We define a following solver circuit \tilde{C} :

Circuit $\tilde{C}^{\Gamma_V^{(g)}, \Gamma_H^{(g)}, hash, C}(x_1, \dots, x_k)$
 Circuit \tilde{C} has good canonical success probability.

Oracle: $\Gamma_V^{(g)}, \Gamma_H^{(k)}, hash, C$
Input: k -wise direct product of puzzles (x_1, \dots, x_k)

Run $C^{(\cdot, \cdot)}(x_1, \dots, x_k)$
If C asks a hint query q **then**
 If $q \in P_{hash}$ **then**
 return \perp
 else
 return $\Gamma_H^{(k)}(q)$ to C

If C asks a verification query on (q, y_1, \dots, y_k) **then**
 If $q \in P_{hash}$ **then**
 return (q, y_1, \dots, y_k)
 else
 answer the verification query with 0
return \perp

Lemma 1.6 *For fixed $P^{(g)}, hash$ the following statement is true*

$$\Pr_{(\pi_1, \dots, \pi_k)} [E^{P^{(g)}, C, hash}(\pi_1, \dots, \pi_k) = 1] \leq \Pr_{(\pi_1, \dots, \pi_k)} [\Gamma_V^{(g)}(\tilde{C}^{\Gamma_V^{(g)}, \Gamma_H^{(g)}, hash}(\pi_1, \dots, \pi_k)) = 1].$$

Proof. We fix the a random bitstring (π_1, \dots, π_k) , $hash$. If C succeeds canonically then

$$\Gamma_V^{(g)}(\tilde{C}^{\Gamma_V^{(g)}, \Gamma_H^{(g)}, hash}(\pi_1, \dots, \pi_k)) = 1.$$

Using this observation, we conclude that

$$\begin{aligned}
\Pr_{(\pi_1, \dots, \pi_k)} [E^{P^{(g)}, C, \text{hash}}(\pi^{(k)}) = 1] &= \sum_{\pi^{(k)} \in \{0,1\}^{kl}} \Pr[E^{P^{(g)}, C, \text{hash}}(\tilde{\pi}^{(k)}) = 1 | \pi^{(k)} = \tilde{\pi}^{(k)}] \Pr[\pi^{(k)} = \tilde{\pi}^{(k)}] \\
&\leq \sum_{\pi^{(k)} \in \{0,1\}^{kl}} \Pr[\Gamma_V^{(g)}(\tilde{C}^{\Gamma_V^{(g)}, \Gamma_H^{(g)}, \text{hash}}(\tilde{\pi}^{(k)})) = 1 | \pi^{(k)} = \tilde{\pi}^{(k)}] \Pr[\pi^{(k)} = \tilde{\pi}^{(k)}] \\
&= \Pr[E^{P^{(g)}, \tilde{C}, \text{hash}}(\pi^{(k)}) = 1]
\end{aligned}$$

□

Algorithm $\text{Gen}(\tilde{C}, g, \varepsilon, \delta, n)$

Oracle: \tilde{C}, g

Input: ε, δ, n

Output: A circuit D

If the number of puzzles to solve equals one **then**
 return \tilde{C}

For $i := 1$ to $\frac{6k}{\varepsilon} \log(n)$
 $\pi^* \leftarrow \{0, 1\}^l$
 $\tilde{S}_{\pi^*, 0} := \text{EvaluateSurplus}(\pi^*, 0)$
 $\tilde{S}_{\pi^*, 1} := \text{EvaluateSurplus}(\pi^*, 1)$
 If $\tilde{S}_{\pi^*, 0} \geq (1 - \frac{3}{4k})\varepsilon$ or $\tilde{S}_{\pi^*, 1} \geq (1 - \frac{3}{4k})\varepsilon$
 $\tilde{C}' := \tilde{C}$ with the first input fixed on π^*
 return $\text{Gen}(\tilde{C}', g, \varepsilon, \delta, n)$
 // all estimates are lower than $(1 - \frac{3}{4k})\varepsilon$
return $D^{\tilde{C}}$

EvaluateSurplus (π^*, b)

For $i := 1$ to N_k
 $(\pi_2, \dots, \pi_k) \xleftarrow{\$} \{0, 1\}^{(k-1)l}$
 $(c_1, \dots, c_k) := \text{EvaluatePuzzles}(\pi^*, \pi_2, \dots, \pi_k)$
 $\tilde{S}_{\pi^*, b}^i := g(b, c_2, \dots, c_k) - \Pr_{(u_2, \dots, u_k)} [g(b, u_2, \dots, u_k) = 1]$
 return $\frac{1}{N_k} \sum_{i=1}^{N_k} \tilde{S}_{\pi^*, b}^i$

EvalutePuzzles $(\pi^{(k)})$

$(x^{(k)}, \Gamma_V^{(g)}, \Gamma_H^{(k)}) := P^{(g)}(\pi^{(k)})$
 For $i := 1$ to k
 $(x_i, \Gamma_V^i, \Gamma_H^i) := P^{(1)}(\pi_i)$
 $(q, y^k) := \tilde{C}^{\Gamma_V^{(g)}, \Gamma_H^{(k)}}(x_1, x_2, \dots, x_k)$
 For $i := 1$ to k
 $c_i := \Gamma_v^i(q, y_i)$
 return (c_1, \dots, c_k)

Circuit $D^{\tilde{C}, P^{(1)}}$

Oracle: A circuit \tilde{C} with the first n puzzles fixed, $P^{(1)}$

Input: A puzzle x^* , a random bitstring $r \in \{0, 1\}^*$

For $i := 1$ to $\frac{6k}{\varepsilon} \log(\frac{6k}{\varepsilon})$
 $\pi^{(k)} \leftarrow \{0, 1\}^{(k-n-1)l}$ //read bits from r
 $(c_1, \dots, c_{k-n-1}) := \text{EvaluatePuzzles}(\pi^{(k-n-1)})$
If $g(1, c_2, \dots, c_k) = 1 \wedge g(0, c_2, \dots, c_k) = 0$
 For $i := 1$ to $k - n - 1$
 $(x_i, \Gamma_V^i, \Gamma_H^i) := P^{(1)}(\pi_i)$
 $(q, y_1, \dots, y_{k-n-1}) := \tilde{C}(x^*, x_2, \dots, x_{k-n-1})$
 return y_1
return \perp

For $k = 1$ the function $g : \{0, 1\} \rightarrow \{0, 1\}$ is either an identity or a constant function. If g is identity then the success probability of \tilde{C} is at least $\delta + \varepsilon$ and \tilde{C} can be directly used to solve a puzzle. In case when g is a constant function the statement is vacuously true.

Let (q, y_1, \dots, y_k) denote the output of \tilde{C} and $c_i := \Gamma_V(q, y_i)$. We define a surplus:

$$S_{\pi^*, b} = \Pr_{\pi^{(k)}}[g(b, c_2, \dots, c_k) = 1] - \Pr_{\mu^{(k)}}[g(b, u_2, \dots, u_k) = 1] \quad (0.0.4)$$

The surplus $S_{\pi^*, b}$ tells us how good \tilde{C} performs when the first puzzle is fixed, and instead c_1 the value b is used. The procedure **EvaluateSurplus** returns the estimate for $\tilde{S}_{\pi^*, b}$. All puzzles used during obtaining the estimate are generated internally. Therefore, it is possible to provide answers for all hint and verification queries. The returned estimate $\tilde{S}_{\pi^*, b}$ differs from $S_{\pi^*, b}$ by at most $\frac{\varepsilon}{4k}$ almost surely. Therefore, if $\tilde{S}_{\pi^*, b} \geq (1 - \frac{3}{4k})\varepsilon$ then $S_{\pi^*, b} \geq (1 - \frac{1}{k})\varepsilon$ almost surely, and we fix the first bit of $g'(b_2, \dots, b_k) := g(b, b_2, \dots, b_k)$, and the first puzzle of \tilde{C} for the one generated from π^* which yields a new circuit \tilde{C}' . The circuit \tilde{C}' satisfies the conditions of Lemma 1.5 and we recurse using \tilde{C}' and the monotone function g' .

If all estimates are less than $(1 - \frac{3}{4k})\varepsilon$, then intuitively \tilde{C} does not perform much better on the remaining $k - 1$ puzzles than an algorithm that solves each puzzle independent with probability δ . However, from the assumption we know that on all k puzzles \tilde{C} has higher success probability. Therefore, it is likely that the first puzzle is correctly solved with probability higher than δ . We now show that this intuition is indeed correct. For a fixed puzzle x^* using (0.0.4), we get

$$\begin{aligned} & \Pr_{u \leftarrow \mu_\delta^k}[g(1, u_2, \dots, u_k) = 1] - \Pr_{u \leftarrow \mu_\delta^k}[g(0, u_2, \dots, u_k) = 1] = \\ & \Pr_{\pi^{(k)}}[g(1, c_2, \dots, c_k) = 1 \mid \pi_1 = \pi^*] - \Pr_{\pi^k}[g(0, c_2, \dots, c_k) = 1 \mid \pi_1 = \pi^*] - (S_{\pi^*, 1} - S_{\pi^*, 0}). \end{aligned} \quad (0.0.5)$$

From the monotonicity of g we know that for any set of tuples (b_1, \dots, b_k) and sets $G_0 = \{(b_1, b_2, \dots, b_k) : g(0, b_2, \dots, b_k) = 1\}$, $G_1 = \{(b_1, b_2, \dots, b_k) : g(1, b_2, \dots, b_k) = 1\}$ we have $G_0 \subseteq G_1$. Hence, we can write (0.0.5):

$$\begin{aligned} & \Pr_{\mu_\delta^k}[g(1, \mu_2, \dots, \mu_k) = 1 \wedge g(0, \mu_2, \dots, \mu_k) = 0] = \\ & \Pr_{\pi^{(k)}}[g(1, c_2, \dots, c_k) = 1 \wedge g(0, c_2, \dots, c_k) = 0 \mid \pi_1 = \pi^*] - (S_{\pi^*, 1} - S_{\pi^*, 0}). \end{aligned} \quad (0.0.6)$$

Let $G_{\mu^{(k)}}$ denote the event $g(1, u_2, \dots, u_k) = 1 \wedge g(0, u_2, \dots, u_k) = 0$, and correspondingly $G_{\pi^{(k)}} := g(1, c_2, \dots, c_k) = 1 \wedge g(0, c_2, \dots, c_k) = 0$. Then multiplying and dividing $\Pr[\Gamma_V^{(g)}(D(x^*, \pi^{(k)})) =$

$1 \mid \pi_1 = \pi^*$] by (0.0.6) we get

$$\begin{aligned} \Pr_r[\Gamma_V^{(g)}(D(x^*, r)) = 1 \mid \pi_1 = \pi^*] &= \frac{\Pr_r[\Gamma_V^{(g)}(D(x^*, r)) = 1 \mid \pi_1 = \pi^*] \Pr_{\pi^{(k)}}[G_\pi \mid \pi_1 = \pi^*]}{\Pr_{u \leftarrow \mu_\delta^k}[G_\mu]} \\ &\quad - \frac{\Pr_r[\Gamma_V^{(g)}(D(x^*, r)) = 1 \mid \pi_1 = \pi^*](S_{\pi^*,1} - S_{\pi^*,0})}{\Pr_{u \leftarrow \mu_\delta^k}[G_\mu]} \end{aligned} \quad (0.0.7)$$

If output of $D(x^*, r) \neq \perp$ then we denote $c_i := \Gamma_V^i(q, y_i)$. We can write the first summand of (0.0.7) as

$$\begin{aligned} \Pr_r[\Gamma_V^{(g)}(D(x^*, r)) = 1 \mid \pi_1 = \pi^*] \Pr_{\pi^{(k)}}[G_\pi \mid \pi_1 = \pi^*] &= \\ \Pr_r[D(x^*, r) \neq \perp \mid \pi_1 = \pi^*] \Pr_{\pi^{(k)}}[c_1 = 1 \mid G_\pi, \pi_1 = \pi^*] \Pr_{\pi^{(k)}}[G_\pi \mid \pi_1 = \pi^*] \end{aligned} \quad (0.0.8)$$

where we make use of the fact that the event G_π implies $D(x^*, r) \neq \perp$. We consider two cases. For $\Pr_{\pi^k}[g(1, c_2, \dots, c_k) = 1 \wedge g(0, c_2, \dots, c_k) = 0 \mid \pi_1 = \pi^*] \leq \frac{\varepsilon}{6k}$ then

$$\Pr_{\pi^{(k)}}[c_1 = 1 \mid G_\pi, \pi_1 = \pi^*] \Pr_{\pi^{(k)}}[G_\pi \mid \pi_1 = \pi^*] \leq \frac{\varepsilon}{6k}, \quad (0.0.9)$$

and when $\Pr_{\pi^k}[g(1, c_2, \dots, c_k) = 1 \wedge g(0, c_2, \dots, c_k) = 0] > \frac{\varepsilon}{6k}$ then circuit D outputs \perp only if it fails in all $\frac{6k}{\varepsilon} \log(\frac{6k}{\varepsilon})$ iterations to find $\pi^{(k)}$ such that $g(1, c_2, \dots, c_k) = 1 \wedge g(0, c_2, \dots, c_k) = 0$ which happens with probability

$$\Pr_r[D(x^*, r) = \perp \mid \pi_1 = \pi^*] \leq (1 - \frac{\varepsilon}{6k})^{\frac{6k}{\varepsilon} \log(\frac{6k}{\varepsilon})} \leq \frac{\varepsilon}{6k}. \quad (0.0.10)$$

We conclude that in both cases:

$$\begin{aligned} \Pr_r[D(x^*, r) \neq \perp \mid \pi_1 = \pi^*] \Pr_{\pi^{(k)}}[c_1 = 1 \mid G_\pi, \pi_1 = \pi^*] \Pr_{\pi^{(k)}}[G_\pi \mid \pi_1 = \pi^*] \\ \geq \Pr_{\pi^{(k)}}[c_1 = 1 \mid G_\pi, \pi_1 = \pi^*] \Pr_{\pi^{(k)}}[G_\pi \mid \pi_1 = \pi^*] - \frac{\varepsilon}{6k}. \end{aligned} \quad (0.0.11)$$

Therefore, we have

$$\begin{aligned} \Pr_r[D(x^*, r) \neq \perp \mid \pi_1 = \pi^*] \Pr_{\pi^{(k)}}[c_1 = 1 \mid G_\pi, \pi_1 = \pi^*] \Pr_{\pi^{(k)}}[G_\pi \mid \pi_1 = \pi^*] \\ = \Pr_{\pi^{(k)}}[c_1 = 1 \wedge g(1, c_2, \dots, c_k) = 1 \wedge g(0, c_2, \dots, c_k) = 0 \mid \pi_1 = \pi^*] - \frac{\varepsilon}{6k} \\ = \Pr_{\pi^{(k)}}[g(c_1, c_2, \dots, c_k) = 1 \wedge g(0, c_2, \dots, c_k) = 0 \mid \pi_1 = \pi^*] - \frac{\varepsilon}{6k} \\ = \Pr_{\pi^{(k)}}[g(c_1, c_2, \dots, c_k) = 1 \mid \pi_1 = \pi^*] - \Pr_{\pi^{(k)}}[g(0, c_2, \dots, c_k) = 0 \mid \pi_1 = \pi^*] - \frac{\varepsilon}{6k}, \end{aligned}$$

and finally by (0.0.4)

$$\begin{aligned} \Pr_r[D(x^*, r) \neq \perp \mid \pi_1 = \pi^*] \Pr_{\pi^{(k)}}[c_1 = 1 \mid G_\pi, \pi_1 = \pi^*] \Pr_{\pi^{(k)}}[G_\pi \mid \pi_1 = \pi^*] \\ = \Pr_{\pi^{(k)}}[g(c_1, c_2, \dots, c_k) = 1 \mid \pi_1 = \pi^*] - \Pr_{\mu_\delta^{(k)}}[g(0, \mu_2, \dots, \mu_k) = 0 \mid \pi_1 = \pi^*] - S_{\pi^*,0} - \frac{\varepsilon}{6k}. \end{aligned} \quad (0.0.12)$$

Inserting this result into the equation (0.0.7) yields

$$\begin{aligned}
\Pr_{r,\pi}[D(x,r) = 1] &= \mathbb{E}_\pi \left[\Pr_r[D(x,r) = 1 \mid \pi_1 = \pi^*] \right] \\
&= \mathbb{E}_\pi \left[\frac{\Pr_{\pi^{(k)}}[g(c) = 1 \mid \pi_1 = \pi^*] - \Pr_{\mu_\delta^{(k)}}[g(0, \mu_2, \dots, \mu_k) = 0 \mid \pi_1 = \pi^*] - \frac{\varepsilon}{6k}}{\Pr_{\mu_\delta^k}[G_\mu]} \right] \\
&\quad - \mathbb{E}_\pi \left[\frac{S_{\pi^*,0} + \Pr_r[\Gamma_V^{(g)}(D(x^*, r)) = 1 \mid \pi_1 = \pi^*](S_{\pi^*,1} - S_{\pi^*,0})}{\Pr_{\mu_\delta^k}[G_\mu]} \right] \tag{0.0.13}
\end{aligned}$$

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

$$\Pr_\pi \left[\left(S_{\pi,0} \leq (1 - \frac{1}{2k})\varepsilon \right) \wedge \left(S_{\pi,1} \leq (1 - \frac{1}{2k})\varepsilon \right) \right] < 1 - \frac{\varepsilon}{6k}, \tag{0.0.14}$$

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

$$\Pr_\pi \left[\left(S_{\pi,0} \leq (1 - \frac{1}{2k})\varepsilon \right) \wedge \left(S_{\pi,1} \leq (1 - \frac{1}{2k})\varepsilon \right) \right] \geq 1 - \frac{\varepsilon}{6k}. \tag{0.0.15}$$

Let us define a set

$$\mathcal{W} = \left\{ \pi : \left(S_{\pi,0} \leq (1 - \frac{1}{2k})\varepsilon \right) \wedge \left(S_{\pi,1} \leq (1 - \frac{1}{2k})\varepsilon \right) \right\} \tag{0.0.16}$$

and use \mathcal{W}^c to denote the complement of \mathcal{W} . We bound the second summand in (0.0.13)

$$\begin{aligned}
&\mathbb{E}_\pi \left[S_{\pi^*,0} + \Pr_r[\Gamma_V^{(g)}(D(x^*, r)) = 1 \mid \pi_1 = \pi^*](S_{\pi^*,1} - S_{\pi^*,0}) \right] \\
&= \mathbb{E}_{\pi \in \mathcal{W}^c} \left[S_{\pi^*,0} + \Pr_r[\Gamma_V^{(g)}(D(x^*, r)) = 1 \mid \pi = \pi^*](S_{\pi^*,1} - S_{\pi^*,0}) \right] \\
&\quad + \mathbb{E}_{\pi \in \mathcal{W}} \left[S_{\pi^*,0} + \Pr_r[\Gamma_V^{(g)}(D(x^*, r)) = 1 \mid \pi = \pi^*](S_{\pi^*,1} - S_{\pi^*,0}) \right] \tag{0.0.17}
\end{aligned}$$

$$\leq \frac{\varepsilon}{6k} + \mathbb{E}_{\pi \in \mathcal{W}^c} \left[S_{\pi^*,0} + \Pr_r[\Gamma_V^{(g)}(D(x^*, r)) = 1 \mid \pi = \pi^*]\left((1 - \frac{1}{2k})\varepsilon - S_{\pi^*,0}\right) \right] \tag{0.0.18}$$

$$\leq \frac{\varepsilon}{6k} + 1 - \frac{\varepsilon}{2k} = 1 - \frac{\varepsilon}{3k} \tag{0.0.19}$$

Finally, we insert this result into equation (0.0.13) and make use of the fact

$$\begin{aligned}
\Pr[g(u) = 1] &= \Pr[(g(0, \mu_2, \dots, \mu_k) = 1) \vee (g(1, \mu_2, \dots, \mu_k) = 1 \wedge g(0, \mu_2, \dots, \mu_k) = 0 \wedge \mu_1 = 1)] \\
&= \Pr[g(0, \mu_2, \dots, \mu_k) = 1] + \Pr[g(1, \mu_2, \dots, \mu_k) = 1 \wedge g(0, \mu_2, \dots, \mu_k) = 0] \Pr[\mu_1 = 1]
\end{aligned}$$

which yields

$$\Pr_{r,\pi}[D(x,r) = 1] \geq \mathbb{E}_\pi \left[\frac{\Pr_{\pi^{(k)}}[g(c) = 1 \mid \pi_1 = \pi^*] - \Pr_{\mu_\delta^{(k)}}[g(0, \mu_2, \dots, \mu_k) = 0] - (1 - \frac{1}{6k})\varepsilon}{\Pr_{\mu_\delta^k}[G_\mu]} \right]$$

Using the assumptions of Lemma 1.5, we get

$$\begin{aligned}
\Pr_{r,\pi}[D(x,r) = 1] &\geq \frac{\Pr_{\mu_\delta^{(k)}}[g(\mu) = 1] + \varepsilon + \Pr_{\mu_\delta^{(k)}}[g(0, \mu_2, \dots, \mu_k) = 0] - (1 - \frac{1}{6k})\varepsilon}{\Pr_{\mu_\delta^k}[G_\mu]} \\
&\geq \frac{\varepsilon + \delta \Pr_{\mu_\delta^{(k)}}[G_\mu] - (1 - \frac{1}{6k})\varepsilon}{\Pr_{\mu_\delta^k}[G_\mu]} \geq \delta + \frac{\varepsilon}{6k}
\end{aligned}$$