We write $u \leftarrow \mu_\delta^k$ to denote a tuple u of length k which each element is independently drawn from the Bernoulli distribution with parameter δ.

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 execution of the algorithm P with the randomness fixed to π. The algorithm P outputs circuits Γ_V, Γ_H and a bitstring $x \in \{0,1\}^*$. The circuit Γ_V takes as input $q \in Q$, an answer $y \in \{0,1\}^*$, and outputs a bit. An answer y is a correct solution of x for q if and only if $\Gamma_V(q, y) = 1$. The circuit Γ_H on input $q \in Q$ outputs a hint such that $\Gamma_V(q, \Gamma_H(q)) = 1$.*

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 execution of S with the input x and the randomness fixed to 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 succeeds if and only if it makes a verification query (q, y) such that $\Gamma_V(q, y) = 1$, and S 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 $P^{(g)}(\pi^{(k)})$ to denote the execution of $P^{(g)}$ with the randomness fixed to $\pi^{(k)} := (\pi_1, \dots, \pi_k)$. The algorithm $P^{(g)}(\pi^{(k)})$ 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 $(x_i, \Gamma_V^i, \Gamma_H^i) := P^{(1)}(\pi_i)$.

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

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 $Q_{Solved} := \{q : \ C^{\Gamma_V,\Gamma_H}$ asked a verification query (q, y) and $\Gamma_V(q, y) = 1\}$
 Let $Q_{Hint} := \{q : C^{\Gamma_V,\Gamma_H}$ asked a hint query on q$\}$
If $\exists q \in Q_{solved} : q \notin Q_{Hint}$ **then**
 return 1
else
 return 0

The success probability of C in the experiment *Success* in solving a puzzle defined by P is

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

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

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

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

then D satisfies almost surely

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

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

The Theorem 1.3 implies that if there is no good solver for a puzzle defined by $P^{(1)}$, then a good solver for a k-wise direct product of $P^{(1)}$ does not exist.

Let C be any solver for $P^{(1)}$ defined as in Definition 1.1. A verification query (q, y) of C for which a hint query on this q has been asked before can not be a successfully verification query. Therefore, without loss of generality we make an assumption that C does not ask verification queries on $q \in Q$, for which a hint query has been asked before.

TODO: Write it more clearly, give more intuition about the function g() and then why we can approach the problem in this way.

The idea of the algorithm *Gen* is to find $k-1$ puzzles and a position for an input puzzle x, such that when C runs with these $k-1$ puzzles and x placed on the right position, then x is often successfully solved. To find such a position for x and $k-1$ puzzles *Gen* runs C repeatedly on different $k-1$ tuples of puzzles. Even if *Gen* finds a set of puzzles and a position for x, such that x is often solved 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 $hash : Q \to \{0, 1, \ldots, 2(h+v)-1\}$, then a set $P_{hash} \subseteq Q$, defined with respect to $hash$, is the set of preimages of 0 for $hash$. The idea is that P_{hash} contains $q \in Q$ on which C is not allowed to ask hint queries and q on which the first successful verification query is asked is in P_{hash}. Therefore, if C makes a verification query on $q \in P_{hash}$ we know that no hint query is ever asked on this q. In the experiment $CanonicalSuccess$ a circuit C succeeds if and only if it ask a verification query on $q \in P_{hash}$ and no hint query is asked on $q \in P_{hash}$. Finally, Lemma 1.4 states that it is possible to find $hash$ such that success probability of C in the experiment $CanonicalSuccess$ is not much worser than in the experiment $Success$.

In the experiment $CanonicalSuccess$ we denote the ith query of C as q_i if it is a hint query, and as (q_i, y_i) if it is a verification query.

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

Oracle: A problem poser P. A solver circuit $C^{(\cdot,\cdot)}$.
A function $hash : Q \leftarrow \{0,\ldots,2(h+v)-1\}$.
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_j,y_j) be the first verification query such that $C^{\Gamma_V,\Gamma_H}(q_j,y_j)=1$, or an arbitrary verification query if C does not succeed.

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

Similarly as for the experiment $Success$, we define the success probability of C with respect to a function $hash$ in the experiment $CanonicalSuccess$ in solving a puzzle defined by P as

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

For fixed $hash$ and $P^{(1)}$ a canonical success of C for $\pi^{(k)},\rho$ is a situation when $CanonicalSuccess^{P^{(g)},C^{(\cdot,\cdot)},hash}(\pi^{(k)},\rho)=1$. We show that if for a fixed $P^{(1)}$ a solver circuit C often succeeds in the experiment $Success$ for $P^{(g)}$, then it also often successful in the experiment $CanonicalSuccess$ for $P^{(g)}$.

Lemma 1.4 ***(Success probability in solving a k-wise direct product of $P^{(1)}$ with respect to a function*** *hash.**)*** *For fixed $P^{(1)}$ let C succeed in the experiment Success for $P^{(g)}$ with probability γ, asking at most h hint queries and v verification queries. There exists a probabilistic algorithm, with oracle access to C and $P^{(g)}$, that runs in time $O((h+v)^4/\gamma^4)$, and with high probability outputs a function hash $: Q \to \{0,\ldots,2(h+v)-1\}$ such that success probability of C with respect to P_{hash} in the experiment CanonicalSuccess is at least $\frac{\gamma}{8(h+v)}$.*

Proof. We fix $P^{(1)}$ and C in the whole proof. Let $\mathcal{H}$ be a family of pairwise independent hash functions $Q \to \{0,1,\ldots,2(h+v)-1\}$. For all $i \neq j \in \{1,\ldots,(h+v)\}$ and $k,l \in \{0,1,\ldots,2(h+v)-1\}$ by pairwise independence property of $\mathcal{H}$, we have

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

Let $\pi^{(k)},\rho$ be fixed. We consider the experiment $CanonicalSuccess$ for $P^{(g)}$. in which we 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 (0.0.3) 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 the fact that $j \leq (h+v)$ 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 $\mathcal{P}_{Success}$ be the set of all $(\pi^{(k)}, \rho)$ for which C succeeds in the random experiment $Success$ for $P^{(g)}$. Furthermore, we denote the set of those $(\pi^{(k)}, \rho)$ for which $CanonicalSuccess^{P^{(g)},C(;),hash}(\pi^{(k)}) = 1$ by $\mathcal{P}_{Canonical}$. For fixed $\pi^{(k)}, \rho$, if C succeeds canonically, then it also succeeds in the experiment $Success$ for $P^{(g)}$. Hence, $\mathcal{P}_{Canonical} \subseteq \mathcal{P}_{Success}$, and we have

$$\begin{aligned}\Pr_{\substack{hash \leftarrow \mathcal{H}\\ \pi^{(k)},\rho}}\left[CanonicalSuccess^{P^{(g)},C^{(\cdot,\cdot)},hash}(\pi^{(k)},\rho)=1\right] &= \mathop{\mathbb{E}}_{(\pi^{(k)},\rho)\in\mathcal{P}_{Success}}\left[\Pr_{hash\leftarrow\mathcal{H}}[X=1]\right]\\ &\geq \frac{\gamma}{4(h+v)}. \qquad (0.0.4)\end{aligned}$$

Algorithm: FindHash

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

For $i = 1$ to $32(h+v)^2/\gamma^2$
    $hash \xleftarrow{\$} \mathcal{H}$
    $count := 0$
    For $j := 1$ to $32(h+v)^2/\gamma^2$
        $\pi^{(k)} \xleftarrow{\$} \{0,1\}^{kl}$
        If $CanonicalSuccess^{P^{(g)},C^{(\cdot,\cdot)},hash}(\pi^{(k)}) = 1$ **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

$$\Pr_{\pi^{(k)},\rho}\left[CanonicalSuccess^{P^{(g)},C^{(\cdot,\cdot)},hash}(\pi^{(k)},\rho)=1\right] \geq \frac{\gamma}{4(h+v)},$$

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

$$\Pr_{\pi^{(k)},\rho}\left[CanonicalSuccess^{P^{(g)},C^{(\cdot,\cdot)},hash}(\pi^{(k)},\rho)=1\right] \leq \frac{\gamma}{8(h+v)}.$$

Additionally, for a fixed $hash$, we define binary random variables $X_1, \ldots, X_N$ 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 $\mathbb{E}_{\pi^{(k)},\rho}[X_i] < \frac{\gamma}{8(h+v)}$. Therefore, for any fixed $hash \in \mathcal{H}_{Bad}$ using the Chernoff bound we get

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

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

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

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 (0.0.4) 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 $K = N = 32(h+v)^2/\gamma^2$. □

We define the following solver circuit $\widetilde{C}$ for a k-wise direct product of $P^{(1)}$.

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

Oracle: $\Gamma_V^{(g)}, \Gamma_H^{(k)}, hash, C$
Input: puzzles $x^{(k)}$, bitstring ρ
Output: A tuple $(q, y_1, \ldots, y_k)$ or $\perp$.

Run $C^{\Gamma_V^{(g)},\Gamma_H^{(k)}}(x^{(k)},\rho)$
  If C asks a hint query on q **then**
    If $q \in P_{hash}$ **then**
      return $\perp$
    else
      answer the query using $\Gamma_H^{(k)}(q)$

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

Lemma 1.5 *For fixed $P^{(1)}$ and hash the following statement is true*

$$\Pr_{\pi^{(k)},\rho}[CanonicalSuccess^{P^{(g)},C,hash}(\pi^{(k)},\rho) = 1] \leq \Pr_{\substack{\pi^{(k)},\rho \\ (x^{(k)},\Gamma_V^{(g)},\Gamma_H^{(k)}) := P^{(g)}(\pi^{(k)})}}[\Gamma_V^{(g)}(\widetilde{C}^{\Gamma_V^{(g)},\Gamma_H^{(k)},hash}(x^{(k)},\rho)) = 1].$$

Proof. We observe that for fixed $\pi^{(k)}, \rho$ if C succeeds canonically, then for $(x^{(k)}, \Gamma_V^{(g)}, \Gamma_H^{(g)} := P^{(g)}(\pi^{(k)}))$ we have

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

Using this observation, we conclude that

$$\begin{aligned}
&\Pr_{\pi^{(k)},\rho}\left[CanonicalSuccess^{P^{(g)},C,hash}(\pi^{(k)},\rho)=1\right] \\
&= \mathop{\mathbb{E}}_{\pi^{(k)},\rho}\left[\Pr\left[CanonicalSuccess^{P^{(g)},C,hash}(\pi^{(k)},\rho)=1\right]\right] \\
&\leq \mathop{\mathbb{E}}_{\pi^{(k)},\rho}\left[\Pr\left[CanonicalSuccess^{P^{(g)},\widetilde{C},hash}(\pi^{(k)},\rho)=1\right]\right] \\
&= \Pr_{\substack{\pi^{(k)},\rho \\ \left(x^{(k)},\Gamma_V^{(g)},\Gamma_H^{(k)}\right):=P^{(g)}(\pi^{(k)})}}[\Gamma_V^{(g)}(\widetilde{C}^{\Gamma_V^{(g)},\Gamma_H^{(k)},hash}(x^{(k)},\rho))=1].
\end{aligned}$$

□

Therefore, from a circuit C we can build a circuit $\widetilde{C}$ that outputs $\perp$ or $(q, y_1, \dots, y_k)$ such that $q \in P_{hash}$. Furthermore, the circuit $\widetilde{C}$ asks no verification queries, and every hint query on q is such that $q \notin P_{hash}$.

TODO: Hash function is taken from Lemma 1.5

Lemma 1.6 ***(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 for $P^{(g)}$, a monotone function $g : \{0,1\}^{(k)} \to \{0,1\}$, a function $hash : Q \to \{0, \dots, 2(h+v)-1\}$, parameters ε, δ, n, 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

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

then D satisfies almost surely

$$\Pr_{\substack{\pi,\sigma \\ (x,\Gamma_V,\Gamma_H):=P^{(1)}(\pi)}}\left[\Gamma_V(D^{P^{(1)},C,\Gamma_V,\Gamma_H,hash}(x,\sigma))=1\right] \geq (\delta + \frac{\varepsilon}{6k}).$$

Additionally, Gen requires only oracle access to g, $P^{(1)}$ and C. Furthermore, $Size(D) \leq Size(C)\frac{6k}{\varepsilon}$ and $Time(Gen) = poly(k, \frac{1}{\varepsilon}, n, v, h)$.

Proof. First we define helper procedures **EvalutePuzzles** and **EvaluateSurplus**.

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

Oracle: A circuit C, an algorithm $P^{(1)}$, a function $hash$.
Input: Bitstrings $\pi^{(k)}$, ρ, an integer k.
Output: A tuple $(c_1, \dots, c_k)$.

$(x^{(k)}, \Gamma_V^{(g)}, \Gamma_H^{(k)}) := P^{(g)}(\pi^{(k)})$
$(q, y^{(k)}) := \widetilde{C}^{\Gamma_V^{(g)},\Gamma_H^{(k)},hash,C}(x^{(k)}, \rho)$

For $i := 1$ to k **do:**

$(x_i, \Gamma_V^i, \Gamma_H^i) := P^{(1)}(\pi_i)$

For $i := 1$ to k **do:**

$c_i := \Gamma_v^i(q, y_i)$

return $(c_1, \ldots, c_k)$

TODO: Figure out N_K
TODO: Get a sample for $\Pr[g(b, \ldots, b) = 1]$

EvaluateSurplus$^{P^{((1))}, C, hash}(\pi^*, b, k)$

Oracle: An algorithm $P^{(1)}$, a circuit C, a function $hash$.
Input: A bistring π^*, a bit b, an integer k.
Output: A circuit D.

For $i := 1$ to N_k

$(\pi_2, \ldots, \pi_k) \xleftarrow{\$} \{0,1\}^{(k-1)n}$

$\rho \xleftarrow{\$} \{0,1\}^*$

$(c_1, \ldots, c_k) :=$ **EvalutePuzzles**$^{P^{(1)}, C, hash}(\pi^*, \pi_2, \ldots, \pi_k, k)$

$\widetilde{S}^i_{\pi^*, b} := g(b, c_2, \ldots, c_k) - \Pr_{(u_2, \ldots, u_k)}[g(b, u_2, \ldots, u_k) = 1]$

return $\frac{1}{N_k} \sum_{i=1}^{N_k} \widetilde{S}^i_{\pi^*, b}$

Circuit $D^{C, P^{(1)}}(x^*, \sigma)$

Oracle: A circuit C, a poser $P^{(1)}$, a function $hash$.
Input: A puzzle x^*, a bitstring $\sigma \in \{0,1\}^*$.
Output: A circuit D.

Let k be the number of input puzzles taken by C.
For $i := 1$ to $\frac{6k}{\varepsilon} \log(\frac{6k}{\varepsilon})$ **do:**

$\pi^{(k)} \leftarrow$ read $k \cdot n$ bits from σ

$(c_1, \ldots, c_k) :=$ **EvaluatePuzzles**$^{P^{(1)}, C, hash}(\pi^{(k)}, k)$

If $g(1, c_2, \ldots, c_k) = 1 \wedge g(0, c_2, \ldots, c_k) = 0$ **then**

For $i := 1$ to k **do:**

$(x_i, \Gamma_V^i, \Gamma_H^i) := P^{(1)}(\pi_i)$

$(q, y_1, \ldots, y_k) := \widetilde{C}(x^*, x_2, \ldots, x_k)$

return (q, y_1)

return $\perp$

Algorithm *Gen*$(C, g, \varepsilon, \delta, n, v, h, hash)$

Oracle: $C, g, hash$
Input: $\varepsilon, \delta, n, v, h$
Output: A circuit D

Let k be the number of input puzzles taken by C.
If $k = 1$ **then**
  return C

For $i := 1$ to $\frac{6k}{\varepsilon}\log(n)$
  $\pi^* \leftarrow \{0,1\}^n$
  $\widetilde{S}_{\pi^*,0} :=$ **EvaluateSurplus**$^{P^{(1)},C,hash}(\pi^*, 0, k)$
  $\widetilde{S}_{\pi^*,1} :=$ **EvaluateSurplus**$^{P^{(1)},C,hash}(\pi^*, 1, k)$
  If $\widetilde{S}_{\pi^*,0} \geq (1 - \frac{3}{4k})\varepsilon$ or $\widetilde{S}_{\pi^*,1} \geq (1 - \frac{3}{4k})\varepsilon$
    $C' := C$ with the first input fixed on x^*
    $g'(b_2, \ldots, b_k) := g(c_1, b_2, \ldots, b_k)$
    return *Gen*$(\widetilde{C}', g', \varepsilon, \delta, n, v, h, hash)$
// all estimates are lower than $(1 - \frac{3}{4k})\varepsilon$
return $D^{\widetilde{C}}$

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

For fixed $\pi^{(k)}, \rho$ let $(x^{(k)}, \Gamma_V^{(g)}, \Gamma_H^{(k)}) := P^{(g)}(\pi^{(k)})$. Additionally, for any i such that $1 \leq i \leq k$ let us the denote $(x_i, \Gamma_V^i, \Gamma_H^i) := P^{(1)}(\pi_i)$. For $(q, y_1, \ldots, y_k) := \widetilde{C}(x^{(k)}, \rho)$ we denote $c_i := \Gamma_V^i(q, y_i)$. We define the surplus:

$$S_{\pi^*,b} = \Pr_{\pi^{(k)}}[g(b, c_2, \ldots, c_k) = 1] - \Pr_{\mu^{(k)}}[g(b, u_2, \ldots, u_k) = 1] \qquad (0.0.5)$$

The surplus $S_{\pi^*,b}$ tells us how good $\widetilde{C}$ performs when the first puzzle is fixed, and the fact whether $\widetilde{C}$ succeeds in solving the puzzle posed by $P^{(1)}(\pi_1)$ is disregarded. Instead, the bit b is used as the first input to g.

The procedure **EvaluateSurplus** returns the estimate $\widetilde{S}_{\pi^*,b}$ for $S_{\pi^*,b}$. All puzzles used during obtaining the estimate are generated internally. Therefore, it is possible to answer all hint and verification queries, without calls to the verification oracles.

Lemma 1.7 *The estimate $\widetilde{S}_{\pi^*,b}$ returned by EvaluteEstimate differs from $S_{\pi^*,b}$ by at most $\frac{\varepsilon}{4k}$ almost surely.*

TODO: Chernoff for the estimate

From Lemma 1.7 we conclude that if $\widetilde{S}_{\pi^*,b} \geq (1 - \frac{3}{4k})\varepsilon$, then $S_{\pi^*,b} \geq (1 - \frac{1}{k})\varepsilon$ almost surely.

Let us assume that *Gen* manages to find an estimate that satisfies $\widetilde{S}_{\pi^*,b} \geq (1 - \frac{3}{4k})\varepsilon$. In this case we define a new monotone function $g'(b_2, \ldots, b_k) := g(b, b_2, \ldots, b_k)$, and a circuit C' which is by fixing the first input of C to x^*, where $(x^*, \Gamma_V^*, \Gamma_H^*) := P^{(1)}(\pi^*)$. The circuit $\widetilde{C}'$ satisfies the conditions of Lemma 1.6 and we recurse using C' and g'.

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 independent with probability δ. However, from the assumption we know that on all k puzzles $\widetilde{C}$ 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 π^* using (0.0.5), we get

$$\begin{aligned}
&\Pr_{u\leftarrow\mu_\delta^k}[g(1,u_2,\dots,u_k)=1] - \Pr_{u\leftarrow\mu_\delta^k}[g(0,u_2,\dots,u_k)=1] = \\
&\quad \Pr_{\pi^{(k)}}[g(1,c_2,\dots,c_k)=1 \mid \pi_1=\pi^*] - \Pr_{\pi^k}[g(0,c_2,\dots,c_k)=1 \mid \pi_1=\pi^*] - (S_{\pi^*,1}-S_{\pi^*,0}).
\end{aligned} \tag{0.0.6}$$

From the monotonicity of g we know that for any set of tuples $(b_1,\dots,b_k)$ and sets $\mathcal{B}_0 = \{(b_1,b_2,\dots,b_k) : g(0,b_2,\dots,b_k)=1\}$, $\mathcal{B}_1 = \{(b_1,b_2,\dots,b_k) : g(1,b_2,\dots,b_k)=1\}$ we have $G_0 \subseteq G_1$. Hence, we can write (0.0.6):

$$\begin{aligned}
&\Pr_{u\leftarrow\mu_\delta^k}[g(1,u_2,\dots,u_k)=1 \wedge g(0,u_2,\dots,u_k)=0] = \\
&\quad \Pr_{\pi^{(k)}}[g(1,c_2,\dots,c_k)=1 \wedge g(0,c_2,\dots,c_k)=0 \mid \pi_1=\pi^*] - (S_{\pi^*,1}-S_{\pi^*,0}).
\end{aligned} \tag{0.0.7}$$

Let $G_{u^{(k)}}$ denote the event $g(1,u_2,\dots,u_k)=1 \wedge g(0,u_2,\dots,u_k)=0$, and correspondingly $G_{\pi^{(k)}} := g(1,c_2,\dots,c_k)=1) \wedge (g(0,c_2,\dots,c_k)=0$. From (0.0.7) we obtain

$$\begin{aligned}
\Pr_r[\Gamma_V(D(x^*,r))=1 \mid \pi_1=\pi^*] &= \frac{\Pr_r[\Gamma_V(D(x^*,r))=1 \mid \pi_1=\pi^*]\Pr_{\pi^{(k)}}[G_\pi \mid \pi_1=\pi^*]}{\Pr_{u\leftarrow\mu_\delta^k}[G_\mu]} \\
&\quad - \frac{\Pr_r[\Gamma_V(D(x^*,r))=1 \mid \pi_1=\pi^*](S_{\pi^*,1}-S_{\pi^*,0})}{\Pr_{u\leftarrow\mu_\delta^k}[G_\mu]}
\end{aligned} \tag{0.0.8}$$

If $D(x^*,r)\neq\perp$ then we denote $c_i := \Gamma_V^i(q,y_i)$. We can write the first summand of (0.0.8) as

$$\begin{aligned}
&\Pr_r[\Gamma_V^{(g)}(D(x^*,r))=1 \mid \pi_1=\pi^*]\Pr_{\pi^{(k)}}[G_\pi \mid \pi_1=\pi^*] = \\
&\quad \Pr_r[D(x^*,r)\neq\perp \mid \pi_1=\pi^*]\Pr_{\pi^{(k)}}[c_1=1 \mid G_\pi,\pi_1=\pi^*]\Pr_{\pi^{(k)}}[G_\pi \mid \pi_1=\pi^*]
\end{aligned} \tag{0.0.9}$$

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

$$\Pr_{\pi^{(k)}}[c_1=1 \mid G_\pi,\pi_1=\pi^*]\Pr_{\pi^{(k)}}[G_\pi \mid \pi_1=\pi^*] \leq \frac{\varepsilon}{6k}, \tag{0.0.10}$$

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

$$\Pr_r[D(x^*,r)=\perp \mid \pi_1=\pi^*] \leq (1-\frac{\varepsilon}{6k})^{\frac{6k}{\varepsilon}\log(\frac{\varepsilon}{6k})} \leq \frac{\varepsilon}{6k}. \tag{0.0.11}$$

We conclude that in both cases:

$$\begin{aligned}
&\Pr_r[D(x^*,r)\neq\perp \mid \pi_1=\pi^*]\Pr_{\pi^{(k)}}[c_1=1 \mid G_\pi,\pi_1=\pi^*]\Pr_{\pi^{(k)}}[G_\pi \mid \pi_1=\pi^*] \\
&\quad \geq \Pr_{\pi^{(k)}}[c_1=1 \mid G_\pi,\pi_1=\pi^*]\Pr_{\pi^{(k)}}[G_\pi \mid \pi_1=\pi^*] - \frac{\varepsilon}{6k}.
\end{aligned} \tag{0.0.12}$$

Therefore, we have

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

and finally by (0.0.5)

$$\begin{aligned}
&\Pr_r[D(x^*,r) \neq \perp \mid \pi_1 = \pi^*] \Pr_{\pi^{(k)}}[c_1 = 1 \mid G_\pi, \pi_1 = \pi^*] \Pr_{\pi^{(k)}}[G_\pi \mid \pi_1 = \pi^*] \\
&\quad = \Pr_{\pi^{(k)}}[g(c_1, c_2, \ldots, c_k) = 1 \mid \pi_1 = \pi^*] - \Pr_{\mu_\delta^{(k)}}[g(0, \mu_2, \ldots, \mu_k) = 0 \mid \pi_1 = \pi^*] - S_{\pi^*,0} - \frac{\varepsilon}{6k}.
\end{aligned} \tag{0.0.13}$$

Inserting this result into the equation (0.0.8) yields

$$\begin{aligned}
\Pr_{r,\pi}[\Gamma_V(D(x,r)) = 1] &= \mathbb{E}_\pi\left[\Pr_r[D(x,r) = 1 \mid \pi_1 = \pi^*]\right] \\
&= \mathbb{E}_\pi\left[\frac{\Pr_{\pi^{(k)}}[g(c) = 1 \mid \pi_1 = \pi^*] - \Pr_{\mu_\delta^{(k)}}[g(0, \mu_2, \ldots, \mu_k) = 0 \mid \pi_1 = \pi^*] - \frac{\varepsilon}{6k}}{\Pr_{\mu_\delta^k}[G_\mu]}\right] \\
&\quad - \mathbb{E}_\pi\left[\frac{S_{\pi^*,0} + \Pr_r[\Gamma_V^{(g)}(D(x^*,r)) = 1 \mid \pi_1 = \pi^*](S_{\pi^*,1} - S_{\pi^*,0})}{\Pr_{\mu_\delta^k}[G_\mu]}\right]
\end{aligned} \tag{0.0.14}$$

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

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

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

$$\Pr_\pi\left[\left(S_{\pi,0} \leq (1 - \frac{1}{2k})\varepsilon\right) \wedge \left(S_{\pi,1} \leq (1 - \frac{1}{2k})\varepsilon\right)\right] \geq 1 - \frac{\varepsilon}{6k}. \tag{0.0.16}$$

Let us define a set

$$\mathcal{W} = \left\{\pi : \left(S_{\pi,0} \leq (1 - \frac{1}{2k})\varepsilon\right) \wedge \left(S_{\pi,1} \leq (1 - \frac{1}{2k})\varepsilon\right)\right\} \tag{0.0.17}$$

and use $\mathcal{W}^c$ to denote the complement of $\mathcal{W}$. We bound the second summand in (0.0.14)

$$\begin{aligned}
&\mathbb{E}_\pi\left[S_{\pi^*,0} + \Pr_r[\Gamma_V^{(g)}(D(x^*,r)) = 1 \mid \pi_1 = \pi^*](S_{\pi^*,1} - S_{\pi^*,0})\right] \\
&\quad = \mathbb{E}_{\pi \in \mathcal{W}^c}\left[S_{\pi^*,0} + \Pr_r[\Gamma_V^{(g)}(D(x^*,r)) = 1 \mid \pi = \pi^*](S_{\pi^*,1} - S_{\pi^*,0})\right] \\
&\qquad + \mathbb{E}_{\pi \in \mathcal{W}}\left[S_{\pi^*,0} + \Pr_r[\Gamma_V^{(g)}(D(x^*,r)) = 1 \mid \pi = \pi^*](S_{\pi^*,1} - S_{\pi^*,0})\right] &(0.0.18) \\
&\quad \leq \frac{\varepsilon}{6k} + \mathbb{E}_{\pi \in \mathcal{W}^c}\left[S_{\pi^*,0} + \Pr_r[\Gamma_V^{(g)}(D(x^*,r)) = 1 \mid \pi = \pi^*]((1 - \frac{1}{2k})\varepsilon - S_{\pi^*,0})\right] &(0.0.19) \\
&\quad \leq \frac{\varepsilon}{6k} + 1 - \frac{\varepsilon}{2k} = 1 - \frac{\varepsilon}{3k} &(0.0.20)
\end{aligned}$$

Finally, we insert this result into equation (0.0.14) 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 1.6, we get

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

□

TO ASK: Is notation $\rho \xleftarrow{\$} \{0,1\}^*$ correct.