Rational Proofs against Rational Verifiers

Keita Inasawa* Kenji Yasunaga†

March 24, 2017

Abstract

Rational proofs, introduced by Azar and Micali (STOC 2012), are a variant of interactive proofs in which the prover is rational, and may deviate from the protocol for increasing his reward. Guo et al. (ITCS 2014) demonstrated that rational proofs are relevant to delegation of computation. By restricting the prover to be computationally bounded, they presented a one-round delegation scheme with sublinear verification for functions computable by log-space uniform circuits with logarithmic depth. In this work, we study rational proofs in which the verifier is also rational, and may deviate from the protocol for decreasing the prover's reward. We construct a three-message delegation scheme with sublinear verification for functions computable by log-space uniform circuits with polylogarithmic depth in the random oracle model.

1 Introduction

Rational proofs, introduced by Azar and Micali [1], are interactive proofs that utilize the rationality of the prover. Specifically, the verifier pays the prover a *reward* based on the transcript of the interaction. The protocol for a function f is designed so that, on a common input x, the receiver can learn $f(x)$ assuming the prover follows the protocol, and the best way for the prover to receive a maximal reward is to follow the protocol. Thus, the verifier can learn $f(x)$ not by verifying the correctness of the computation by the prover, but by relying on the rationality of the prover.

Azar and Micali [1] demonstrated the power of rational proofs by presenting a one-round rational proof for any search problem in $\#P$. In subsequent work, Azar and Micali [2] gave "super-efficient" rational proofs, in which the verifier runs in logarithmic time, and showed that they capture the class of constant-depth polynomial-size circuits with threshold gates. Guo, Hubáček, Rosen, and Vald [7] introduced the notion of *rational arguments*, where the prover is restricted to be computationally bounded. They constructed a one-round rational argument with a polylogarithmic verification for the class NC^1, of search problems computable by log-space uniform circuits of logarithmic depth. Recently, Guo, Hubáček, Rosen, and Vald [8] gave a construction of a one-round scheme with sublinear verification for all languages in P.

The work of [2, 7, 8] can be seen as *delegation of computation* in the setting of rational proofs. A small device with limited computing power (the verifier) delegates a computation to a cloud server (the prover). On common input x, the device expects the server to return the correct value $f(x)$ by paying a reward for the computation to the server. As far as the server prefers to maximize

*Kanazawa University. Kanazawa, Japan. E-mail: ina@stu.kanazawa-u.ac.jp
†Kanazawa University. Kanazawa, Japan. E-mail: yasunaga@se.kanazawa-u.ac.jp

the reward, the device can learn the correct value. Compared to the previous setting of delegation schemes [6, 10, 9], delegation schemes in rational proofs achieve *polylogarithmic* verification.

In existing rational proofs, the rationality is considered only for the prover. Since the verifier pays a reward to the prover, it seems natural to consider *rational* verifiers. Namely, the small device may try to *minimize* the reward as far as the correct value can be received.

1.1 This Work

We study rational proofs in which verifiers also behave rationally. First, we demonstrate that a rational verifier can indeed reduce the reward by deviating from the protocol of [7], which is for functions computable by threshold circuits. Since the protocol is a *public-coin* protocol, the verifier only samples random coins and sends the result to the prover in each round. The result is used for choosing a child gate of the current gate from the root to the inputs. We observe that if the underlying threshold has a gate in which an input wire is connected with an input to the circuit, the verifier can reduce the reward by intentionally choosing that wire. Although this is not possible for *layered* circuits, we show that even for layered circuits the verifier can reduce the reward by carefully choosing gates for which the expected reward becomes smaller.

Next, we define the notion of *fully-rational* proofs, in which both the prover and the receiver behave rationally, and the protocol can be performed between them. In other words, a protocol for fully-rational proofs assures that following the protocol description is a *Nash equilibrium*. The definition is based on that of rational arguments, introduced in [7]. We present a fully-rational argument for functions computable by threshold circuits. The protocol is a variant of the protocol of [7] for threshold circuits in which random coins of the verifier are chosen by *collective coin-flipping*. Although coin-flipping can be implemented with commitment schemes, they usually require polynomial-time verification, which is not suitable for delegation of computation. Instead, we use a commitment scheme in the random oracle model for achieving polylogarithmic verification. By employing such commitment, we construct a *three*-message protocol for any threshold circuits, while the round complexity of the underlying protocol [7] is d for threshold circuits with depth d. We show that our protocol is a fully-rational proof with polylogarithmic verification for threshold circuits with polylogarithmic depth.

1.2 Related Work

As described above, rational proofs were introduced by Azar and Micali [1]. Rational proofs with logarithmic verification were also studied by Azar and Micali [2]. Guo et al. [7] introduced rational arguments, and showed a one-round protocol with polylogarithmic verification for the class NC^1. Later, Guo et al. [8] presented a one-round rational argument with polylogarithmic verification for all languages in P. Campanelli and Gennaro [4] studied the composability of rational proofs and presented a sequentially composable rational proof for arithmetic circuits. Chen, McCauley, and Singh [5] studied rational proofs that allows multiple provers, and showed that multiple rational provers are strictly more powerful than one.

2 Preliminaries

For $n \in \mathbb{N}$, let $[n] = \{1, 2, \ldots, n\}$. A function $\varepsilon : \mathbb{N} \to \mathbb{R}_{\geq 0}$ is said to be *negligible* if for any polynomial $p(\cdot)$, $\varepsilon(n) < p^{-1}(n)$ for every sufficiently large $n \in \mathbb{N}$. We denote by $\mathsf{negl}(\cdot)$ a negligible

function.

2.1 Rational Proofs

In rational proofs, the verifier pays the prover a reward according to the transcript of the communication. Protocols are designed so that the correct evaluation $f(x)$ can be obtained from the transcript that maximizes the expected reward. For a pair of interactive Turing machines P and V, we denote by $(P,V)(x)$ the random variable representing the transcript between P and V when interacting on common input x. Let $\mathsf{Out}((P,V)(x))$ denote the output of V after interacting with P on common input x. For a reward function Rew that maps transcripts to real values, we denote by $\mathsf{Rew}((P,V)(x))$ the reward calculated by V in the interaction with P on input x.

Definition 1 (Rational Proof)**.** *A function $f : \{0,1\}^* \to \{0,1\}^*$ admits a* rational proof *if there exists a protocol (P,V) and a reward function $\mathsf{Rew} : \{0,1\}^* \to \mathbb{R}_{\geq 0}$ such that for any $x \in \{0,1\}^*$,*

1. $\Pr[\mathsf{Out}((P,V)(x)) = f(x)] = 1$.
2. *For any prover P^*,*
$$E[\mathsf{Rew}((P^*,V)(x))] \leq E[\mathsf{Rew}((P,V)(x))].$$
3. *For any prover P^*, if there is a polynomial $p(\cdot)$ such that*
$$\Pr[\mathsf{Out}((P^*,V)(x)) \neq f(x)] \geq p^{-1}(|x|),$$
then there is a polynomial $q(\cdot)$ such that
$$E[\mathsf{Rew}((P^*,V)(x)] \leq E[\mathsf{Rew}((P,V)(x)] - q^{-1}(|x|).$$

A *public-coin* protocol is one in which every message from V consists of all random coins tossed by V in the round.

2.2 Scoring Rules

The main technical tools employed in the constructions of rational proofs in [1, 2, 7] are *scoring rules*, which can be used for forecasters to report the true weather forecast. A scoring rule assigns a real value $S(Q,\omega)$ to a probability distribution Q and an event ω drawn from the actual distribution P. The expected value is maximized if the forecaster reports the true distribution P as a forecast.

Definition 2 (Strictly Proper Scoring Rule)**.** *Let P be a probability distribution over a probability space Ω. We say that a function $S : \{0,1\}^* \to \mathbb{R}$ is a* strictly proper scoring rule *with respect to P if for every probability distribution $Q \neq P$ over Ω,et*
$$\sum_{\omega \in \Omega} P(\omega)S(P,\omega) > \sum_{\omega \in \Omega} P(\omega)S(Q,\omega).$$

The study of scoring rules was initiated by Brier [3]. A variant of the scoring rule given in [3] is a function
$$S_B(P,\omega) = 2P(\omega) - \sum_{\omega \in \Omega} P(\omega)^2 - 1.$$

2.3 Protocol of Guo et al. [7]

We review the rational proof proposed by Guo et al. [7] for functions computable by log-space uniform threshold circuits. The protocol consists of rational proofs for threshold gates. First, a rational proof for the output gate is performed, which begins with sending the output value to the verifier. Since the reward is calculated by randomly choosing an input wire to the gate, the verifier chooses a child gate uniformly at random. Then, a rational proof for the chosen gate will be performed until an input wire to the circuit is chosen as a child.

Let $f : \{0,1\}^* \to \{0,1\}$ be a function computable by a threshold circuit of depth $d = d(n)$ on input $x \in \{0,1\}^n$. The protocol (P, V) is specified as follows.

1. P: Evaluate the circuit on $x \in \{0,1\}^n$ and send the output value y_1 to V.
2. V: Set $\gamma = 1/(1 + 2m_0^2)$, where m_0 is the largest fan-in over all gates in the circuit.[1] Identify the root gate g_1 and invoke the procedure $\mathsf{Round}(1, g_1, y_1)$,

where $\mathsf{Round}(i, g_i, y_i)$ for $1 \le i \le d$ is defined as follows.

1. V: Choose a child g_{i+1} of g_i uniformly at random.
   - If g_{i+1} is a threshold gate, ask P for the output value of g_{i+1}.
   - Otherwise, if g_{i+1} is an input to the circuit of value $b \in \{0,1\}$, then pay P the reward $\gamma^{d-i}/2 \cdot \mathsf{Brier}(g_i, y_i)$ and halt the protocol, where $\mathsf{Brier}(g, y)$ is defined to be

$$\mathsf{Brier}(g, y) = \begin{cases} 2\beta_y - (1-\beta_y)^2 - \beta_y^2 + 1 & \text{if } b = 1, \\ 2(1-\beta_y) - (1-\beta_y)^2 - \beta_y^2 + 1 & \text{otherwise,} \end{cases}$$

   $\beta_1 = t/m, \beta_0 = (t-1)/m$, and t and m are the threshold and the fan-in of gate g, respectively.
2. P: Send the output value y_{i+1} of the gate g_{i+1} to V.
3. V: Pay P the reward $\gamma^{d-i}/2 \cdot \mathsf{Brier}(g_i, y_i)$ for $b = y_{i+1}$ and invoke $\mathsf{Round}(i+1, g_{i+1}, y_{i+1})$.

In the protocol, Brier's scoring rule is employed such that the prover can choose either β_1 and β_0 as a forecast. For each forecast, the true distribution is $\beta = \#\{\text{input wires with value } 1\}/m$. Brier's rule has the property that the expected score gets higher by answering $y = 1$ if $\beta \ge t/m$, and $y = 0$ if $\beta < t/m$. Thus, the prover has an incentive to answer the true output value for each gate. The scores are less discounted on the consecutive levels from the root to the input in order to prevent the prover from deviating from the protocol at the later stages.

The following theorem holds [7].

Theorem 1 ([7]). *If $f : \{0,1\}^* \to \{0,1\}$ is computable by a family of $O(\log S(n))$-space uniform threshold circuits of size $S(n)$ and depth $d(n)$, then f admits a rational proof such that (1) a protocol (P, V) is a $d(n)$-round public-coin protocol; (2) the communication complexity of P is $d(n)$; (3) the running time of V is $O(d(n) \cdot \mathrm{poly}(\log S(n)))$.*

[1] We can set m_0 to be any number larger than the maximal fan-in.

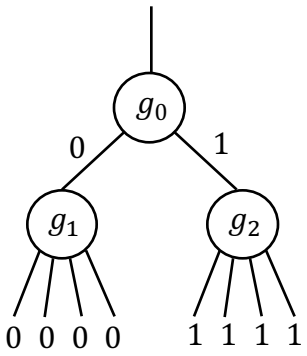

Figure 1: A circuit with three gates

3 Rational Proofs against Rational Verifiers

In rational proofs, the verifier is assumed to follow the protocol honestly. However, the verifier needs to pay a reward to the prover by performing the protocol. It seems natural to consider *rational* verifiers who try to reduce the amount of rewards paid to the prover.

3.1 Behavior of Rational Verifiers

We show that rational verifiers can indeed reduce the expected reward by deviating from the protocol of [7], which is presented in Section 2.3. Note that the verifier can choose next child at each gate in the protocol, where the verifier is supposed to choose uniformly at random.

One trivial example is to choose an input wire to the circuit when other wires are connected to other gates. Since the total score is a sum of scores of chosen gates, the total gets lower if the verifier intentionally chooses an input wire so early.

Although the first example can be avoided by considering *layered* circuits, there is another less trivial example. Consider a gate g with threshold t and fan-in m. Let $(x_1, \ldots, x_m) \in \{0,1\}^m$ be the input to g, and $X = \Pr_{r\in[m]}[x_r = 1]$. The expected reward when the prover answers y as the output value for g is

$$\begin{aligned}\mathbb{E}[\mathsf{Brier}(g,y)] &= X(2\beta_y - (1-\beta_y)^2 - \beta_y^2 + 1) + (1-X)(2(1-\beta_y) - (1-\beta_y)^2 - \beta_y^2 + 1) \\ &= 2(2\beta_y - 1)X + 2 - 2\beta_y^2.\end{aligned}$$

Thus, the expected reward is minimized by choosing larger X if $\beta_y < 1/2$, and choosing smaller X if $\beta_y \geq 1/2$.

For a concrete example, let consider a circuit with three gates g_0, g_1, g_2 as in Figure 1. Suppose that thresholds of g_1 and g_2 are 2 and 1, respectively, and that all the input values to g_1 are 0 and those to g_2 are 1. Then, the left gate g_1 has the values $\beta_0 = 1/4$ and $X = 0$, and the right gate g_2 has the values $\beta_1 = 1/4$ and $X = 1$. Thus, the expected reward becomes smaller if the verifier chooses g_2 since we know that if $\beta_y < 1/2$, the reward is minimized by choosing larger X. Indeed, the reward obtained by choosing g_2 is $7/8$, while that obtained by g_1 is $15/8$.

As illustrated in the above, the verifier can deviate from the protocol so that the expected reward will be smaller than that obtained by following the protocol.

3.2 Definition

We introduce the notion of *fully-rational proof*, in which the protocol can be performed between a rational prover and a rational verifier. Intuitively, it is a rational proof in which the verifier cannot decrease the reward by deviating from the protocol.

The definition is based on the notion of *rational arguments*, introduced in [7], that are a variant of rational proofs in which the prover is restricted to be computationally bounded.

Definition 3 (Fully-Rational Argument)**.** *A function $f : \{0,1\}^* \to \{0,1\}^*$ admits a* fully-rational argument *if there exists a protocol (P, V) and a reward function $\mathsf{Rew} : \{0,1\}^* \to \mathbb{R}_{\geq 0}$ such that for any $x \in \{0,1\}^*$,*

1. $\Pr[\mathsf{Out}((P,V)(x)) = f(x)] = 1$.
2. *For any P^* of size* $\mathrm{poly}(|x|)$*,*
$$E[\mathsf{Rew}((P^*,V)(x))] \leq E[\mathsf{Rew}((P,V)(x))] + \mathsf{negl}(|x|).$$
3. *For any P^* of size* $\mathrm{poly}(|x|)$*, if there is a polynomial $p(\cdot)$ such that*
$$\Pr[\mathsf{Out}((P^*,V)(x)) \neq f(x)] \geq p^{-1}(|x|),$$
then there is a polynomial $q(\cdot)$ such that
$$E[\mathsf{Rew}((P^*,V)(x)] \leq E[\mathsf{Rew}((P,V)(x)] - q^{-1}(|x|).$$
4. *For any V^* of size* $\mathrm{poly}(|x|)$*,*
$$E[\mathsf{Rew}((P,V^*)(x)] \geq E[\mathsf{Rew}((P,V)(x)] - \mathsf{negl}(|x|).$$

The second and forth conditions imply that the strategy of following the protocol is a *computational Nash equilibrium*. In addition, the third condition partially assures a computational *strict* Nash equilibrium, since it guarantees that any deviation by the prover that results in an incorrect answer for $f(x)$ must decrease the reward by a noticeable amount.

4 Our Protocol

We present a protocol for a fully-rational proof based on the protocol of Guo et al. [7]. The problem of rational verifiers in the protocol of [7] is that they can choose next child for minimizing the reward. To prevent such deviation, we use a collective coin-flipping to choose a next child. A coin-flipping protocol can be constructed with a commitment scheme. However, usual commitment schemes do not allow $\mathrm{poly}(\log n)$-time verification. In order to achieve $\mathrm{poly}(\log n)$-time computation for verifiers, we employ a commitment scheme in the random oracle model. Furthermore, the commitment in the random oracle model allows us to achieve a three-message protocol for any threshold circuit, while the round complexity of the underlying protocol [7] is d for threshold circuits with depth d.

Let $f : \{0,1\}^* \to \{0,1\}$ be a function computable by a threshold circuit of depth $d = d(n)$ on input $x \in \{0,1\}^n$. For simplicity, we assume that every fan-in of gates in the circuit is $m = m(n)$. We define our protocol (P, V) using a random oracle $H : [M] \to [M]$, where $M = M(n)$ is a multiple of m satisfying $\omega(\log n) \leq \log M(n) \leq \mathrm{poly}(\log n)$.

1. P : Choose $a \in [M]$ uniformly at random and compute $a_{d-i} = H^i(a)$ for $0 \le i \le d-1$ by querying the random oracle. Evaluate the circuit on $x \in \{0,1\}^n$ and send the output value y_1 and $H^d(a)$ to V.

2. V : Given y_1 and $\tilde{h}$, set $\gamma = 1/(1+2m^2)$ and choose $b_1, \ldots, b_d \in [M]$ uniformly at random and send them to P.

3. P : For $1 \le i \le d$, do the following:

   - Set $r_i = a_i + b_i \bmod m$ and choose the r_i-th child g_{i+1} of the gate g_i, where g_1 is the root gate.
     - If g_{i+1} is a threshold gate, set y_{i+1} to be the output value of g_{i+1} and go to next i.
     - Otherwise, if g_{i+1} is an input to the circuit of value $b \in \{0,1\}$, then set $y_{i+1} = b$ and $y_j = \perp$ for all $i+2 \le j \le d$. Send $a, y_2, \ldots, y_d$ to V.

4. V : Given $a, y_2, \ldots, y_d$ from P, compute $a_{d-i} = H^i(a)$ for $0 \le i \le d-1$.

   - If $H^d(a) \neq \tilde{h}$, then pay nothing to P and halt the protocol.
   - Otherwise, for $1 \le i \le d$, set $r_i = a_i + b_i \bmod m$, identify the r_i-th child g_{i+1} of the gate g_i, and set $\psi_i = \gamma^{d-i}/2 \cdot \mathsf{Brier}(g_i, y_i)$ for $b = y_{i+1}$, where $\mathsf{Brier}(g, y)$ is the same as described in Section 2.3 and $\psi_i = 0$ if $y_{i+1} = \perp$. Pay P the reward $\sum_{j \in [d]} \psi_j$ and halt the protocol.

In the above protocol, we use $H(a_i)$ as a commitment of the value a_i, which is defined to be $a_i = H^{d-i}(a)$ and is uniformly distributed. Since $H^d(a)$ can work as a commitment of the values $a_1, \ldots, a_d$, the prover cannot change the values $a_1, \ldots, a_d$ after sending $H^d(a)$. If the commitment verification fails, the verifier pay nothing to the prover. Since $H^d(a)$ reveals no information about $a_1, \ldots, a_d$, the verifier cannot control the values $r_i = a_i + b_i \bmod m$, which are uniformly distributed. We use these values $r_1, \ldots, r_d$ to choose next children from the root.

Theorem 2. *If $f : \{0,1\}^* \to \{0,1\}$ is computable by a family of $O(\log S(n))$-space uniform threshold circuits of size $S(n)$ and depth $d(n)$, then f admits a fully-rational argument such that (1) a protocol (P, V) is a three-message public-coin protocol; (2) the communication complexity of P is $2 \log M(n) + d(n)$; (3) the running time of V is $O(d(n) \log M(n) + \mathrm{poly}(\log S(n)))$.*

Proof. Our protocol described above is a three-message public-coin protocol. Let $S = S(n)$, $M = M(n)$, and $d = d(n)$. It is not difficult to see that the communication complexity of P is $2 \log M + d$. In the original protocol [7], the reward is computed in time $\mathrm{poly}(\log n)$. Thus, the reward in our protocol is also computed in time $\mathrm{poly}(\log n)$. In addition, the verifier needs to sample d instances from $[M]$, add elements in $[M]$ d times, and identify children d times. Since the circuit is $O(\log S)$-space uniform, identifying a child can be computed in time $\mathrm{poly}(\log S)$. Hence, the running time of the verifier is $O(d(\log M + \mathrm{poly}(\log S))$.

To prove the four properties of fully-rational argument, it is helpful to define two intermediate protocols (P_1, V_1) and (P_2, V_2). The protocol (P_1, V_1) is a variant of the protocol of [7] in which, a sum of two random values $r_i = a_i + b_i \pmod m$ is used for choosing a random child, and the prover sends a commitment $H(a_i)$ of a_i before the receiver sends b_i. The following is a formal description of (P_1, V_1), where the differences are highlighted with underlines.

Protocol (P_1, V_1) **:**

1. P_1: Choose $a_1 \in [M]$ uniformly at random. Evaluate the circuit on $x \in \{0,1\}^n$ and send the output value y_1 and $H(a_1)$ to V_1.

2. V_1: Given y_1 and $\tilde{h}_1$, set $\gamma = 1/(1+2m^2)$. Identify the root gate g_1 and invoke the procedure $\mathsf{Round}_1(1, g_1, y_1)$,

where $\mathsf{Round}_1(i, g_i, y_i)$ for $1 \leq i \leq d$ is defined as follows.

1. V_1: Choose $b_i \in [M]$ uniformly at random, and send b_i to P_1.

2. P_1: Set $r_i = a_i + b_i \bmod m$, and choose the r_i-th child g_{i+1} of g_i.
   If g_{i+1} is a threshold gate, set y_{i+1} to be the output value of g_{i+1}. Otherwise set $y_{i+1} = \perp$.
   Send a_i, $H(a_{i+1})$, and y_{i+1} to V_1.

3. V_1: Given $\tilde{a}_i$, $\tilde{h}_{i+1}$, and y_{i+1}, if $H(\tilde{a}_i) \neq \tilde{h}_i$, pay P_1 nothing and halt the protocol.
   Otherwise, set $r_i = \tilde{a}_i + b_i \bmod m$, and choose the r_i-th child g_{i+1} of g_i.

   - If g_{i+1} is a threshold gate, pay P_1 the reward $\gamma^{d-i}/2 \cdot \mathsf{Brier}(g_i, y_i)$ for $b = y_{i+1}$, and invoke $\mathsf{Round}_1(i+1, g_{i+1}, y_{i+1})$.
   - Otherwise, if g_{i+1} is an input to the circuit of value $b \in \{0,1\}$, then pay P_1 the reward $\gamma^{d-i}/2 \cdot \mathsf{Brier}(g_i, y_i)$ and halt the protocol.

We show that the protocol (P_1, V_1) satisfies the four properties of fully-rational arguments. The first property of completeness immediately follows from the protocol. Let P^* be a prover of size $\mathrm{poly}(|x|)$. It is not difficult to see that as far as r_i's are uniformly distributed, the expected reward is the same as that in the original protocol of [7]. The only way to increase the reward by P^* is to choose $a'_i \in [M]$ satisfying $H(a_i) = H(a'_i)$ after sending a commitment $H(a_i)$. Since M is of size $2^{\omega(\log |x|)}$ and $H : [M] \to [M]$ is a random oracle, it is difficult for P^* to find a collision a'_i except with a negligible probability. Thus, we have that $E[\mathsf{Rew}((P^*, V)(x))] \leq E[\mathsf{Rew}((P, V)(x))] + \mathsf{negl}(|x|)$. To prove the third property, let consider P^* of size $\mathrm{poly}(|x|)$ such that $\Pr[\mathsf{Out}((P^*, V)(x)) \neq f(x)] \geq p^{-1}(|x|)$ for some polynomial $p(\cdot)$. This is the case that P^* sends an incorrect value y_1 at the first step. If so, as in the above analysis, P^* cannot increase the reward by more than a negligible amount. It follows from the third property of a rational proof for the original protocol that there is a polynomial $q(\cdot)$ such that $E[\mathsf{Rew}((P^*, V)(x)] \leq E[\mathsf{Rew}((P, V)(x)] - q^{-1}(|x|)$. Finally, let consider V^* of size $\mathrm{poly}(|x|)$. As far as r_i's are uniformly distributed, V^* cannot decrease the expected reward. Thus, to decrease the reward, V^* needs to learn the information about a_i from $H(a_i)$ before choosing b_i. However, since $H : [M] \to [M]$ is a random oracle with $M = 2^{\omega(\log |x|)}$, the information cannot be learned by any polynomial-time computation except with a negligible probability. Therefore, we have that $E[\mathsf{Rew}((P, V^*)(x)] \geq E[\mathsf{Rew}((P, V)(x)] - \mathsf{negl}(|x|)$.

Next, we define a protocol (P_2, V_2) that is a variant of (P_1, V_1) in which $H(a_{i+1})$ is used as a_i, and thus the prover only needs to sample a random element a and set $a_{d-i} = H^i(a)$. The following is a formal descriptions of (P_2, V_2), where the differences are highlighted with underlines.

Protocol (P_2, V_2) **:**

1. P_2: Choose $a \in [M]$ uniformly at random and compute $a_{d-i} = H^i(a)$ for $0 \leq i \leq d-1$. Evaluate the circuit on $x \in \{0,1\}^n$ and send the output value y_1 and $H(a_1)$ to V_2.
2. V_2: Given y_1 and $\tilde{h}_1$, set $\gamma = 1/(1+2m^2)$. Identify the root gate g_1 and invoke the procedure $\mathsf{Round}_2(1, g_1, y_1)$,

where $\mathsf{Round}_2(i, g_i, y_i)$ for $1 \leq i \leq d$ is defined as follows.

1. V_2: Choose $b_i \in [M]$ uniformly at random, and send b_i to P_2.
2. P_2: Set $r_i = a_i + b_i \bmod m$, and choose the r_i-th child g_{i+1} of g_i. If g_{i+1} is a threshold gate, set y_{i+1} to be the output value of g_{i+1}. Otherwise set $y_{i+1} = \perp$. Send a_i (which is equal to $H(a_{i+1})$) and y_{i+1} to V_2.
3. V_2: Given $\tilde{a}_i$ and y_{i+1}, if $H(\tilde{a}_i) \neq \tilde{h}_i$, pay P_2 nothing and halt the protocol. Otherwise, set $r_i = \tilde{a}_i + b_i \bmod m$, and choose the r_i-th child g_{i+1} of g_i.
   - If g_{i+1} is a threshold gate, pay P_2 the reward $\gamma^{d-i}/2 \cdot \mathsf{Brier}(g_i, y_i)$ for $b = y_{i+1}$, and invoke $\mathsf{Round}_2(i+1, g_{i+1}, y_{i+1})$.
   - Otherwise, if g_{i+1} is an input to the circuit of value $b \in \{0,1\}$, then pay P_2 the reward $\gamma^{d-i}/2 \cdot \mathsf{Brier}(g_i, y_i)$ and halt the protocol.

This protocol also satisfies the four properties of fully-rational arguments. The only difference from (P_1, V_1) is the way of choosing a_i's. Since a_i is chosen as $H(a_{i+1})$ and H is a random oracle, a_i is uniformly distributed except with a negligible probability. Thus, all the properties of fully-rational arguments are satisfied.

Finally, we show that our protocol (P, V) satisfies the properties of fully-rational arguments based on the fact that (P_2, V_2) satisfies them. In the protocol (P, V), the prover does not send $H^i(a_{i+1})$ for $1 \leq i \leq d-1$ before sending $a = a_d$ at the final step of P. As in the previous analysis, a polynomial-size P^* cannot find any collision in H except with a negligible probability. Thus, P^* cannot increase the reward more than a negligible amount in our protocol, which implies the second and the third properties. Since $H^d(a)$ does not reveal any information about $a, H(a), \ldots, H^{d-1}(a)$ for any polynomial-size V^* except with a negligible probability, such V^* cannot decrease the reward by more than a negligible amount, which implies the forth property. Therefore, the statement follows. □

Theorem 2 implies that our protocol achieves $\mathrm{poly}(\log n)$-time verification for functions computable by polynomial-size threshold circuits with $\mathrm{poly}(\log n)$ depth. Although we have assumed that every fan-in of gates in the circuit was m, the restriction can be removed. It is well-known that threshold gates can be simulated with bounded fan-in AND and OR gates, and bounded fan-in AND and OR gates can be simulated with bounded fan-in threshold gates. Thus, any polynomial-size threshold circuit can be simulated with a polynomial-size threshold circuit consisting of bounded fan-in threshold gates.

5 Conclusions

We have shown that the verifier in rational proofs may have an incentive to deviate from the protocol for decreasing the reward for the prover. The notion of fully-rational proof is defined so that the strategy of following the protocol is a Nash equilibrium for rational prover and verifier. We have presented a three-message fully-rational proof with polylogarithmic verification for functions computable by circuits with polylogarithmic depth in the random oracle model. One possible future work is to construct a fully-rational proof with sublinear verification in the standard model. Another one is to reduce the round complexity of our protocol, or to prove the optimality of three-message protocols.

Acknowledgements

This work was supported in part by JSPS/MEXT Grant-in-Aid for Scientific Research Numbers 24240001, 15H00851, and 16H01705.

References

[1] P. D. Azar and S. Micali. Rational proofs. In H. J. Karloff and T. Pitassi, editors, *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012*, pages 1017–1028. ACM, 2012.

[2] P. D. Azar and S. Micali. Super-efficient rational proofs. In M. Kearns, R. P. McAfee, and É. Tardos, editors, *ACM Conference on Electronic Commerce, EC'13*, pages 29–30. ACM, 2013.

[3] G. W. Brier. Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 78(1):1–3, 1950.

[4] M. Campanelli and R. Gennaro. Sequentially composable rational proofs. In M. H. R. Khouzani, E. A. Panaousis, and G. Theodorakopoulos, editors, *Decision and Game Theory for Security - 6th International Conference, GameSec 2015*, volume 9406 of *Lecture Notes in Computer Science*, pages 270–288. Springer, 2015.

[5] J. Chen, S. McCauley, and S. Singh. Rational proofs with multiple provers. In M. Sudan, editor, *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science, ITCS'16*, pages 237–248. ACM, 2016.

[6] S. Goldwasser, Y. T. Kalai, and G. N. Rothblum. Delegating computation: Interactive proofs for muggles. *J. ACM*, 62(4):27:1–27:64, 2015.

[7] S. Guo, P. Hubáček, A. Rosen, and M. Vald. Rational arguments: single round delegation with sublinear verification. In M. Naor, editor, *Innovations in Theoretical Computer Science, ITCS'14*, pages 523–540. ACM, 2014.

[8] S. Guo, P. Hubáček, A. Rosen, and M. Vald. Rational sumchecks. In E. Kushilevitz and T. Malkin, editors, *Theory of Cryptography - 13th International Conference, TCC 2016-A, Proceedings, Part II*, volume 9563 of *Lecture Notes in Computer Science*, pages 319–351. Springer, 2016.

[9] Y. T. Kalai, R. Raz, and R. D. Rothblum. How to delegate computations: the power of no-signaling proofs. In D. B. Shmoys, editor, *Symposium on Theory of Computing, STOC 2014*, pages 485–494. ACM, 2014.

[10] G. N. Rothblum, S. P. Vadhan, and A. Wigderson. Interactive proofs of proximity: delegating computation in sublinear time. In D. Boneh, T. Roughgarden, and J. Feigenbaum, editors, *Symposium on Theory of Computing Conference, STOC 2013*, pages 793–802. ACM, 2013.