

Rational Proofs

Pablo Azar
MIT, CSAIL
Cambridge, MA 02139, USA
azar@csail.mit.edu

Silvio Micali
MIT, CSAIL
Cambridge, MA 02139, USA
silvio@csail.mit.edu

ABSTRACT

We study a new type of proof system, where an unbounded prover and a polynomial time verifier interact, on inputs a string x and a function f , so that the Verifier may learn $f(x)$. The novelty of our setting is that there no longer are “good” or “malicious” provers, but only *rational* ones. In essence, the Verifier has a budget c and gives the Prover a reward $r \in [0, c]$ determined by the transcript of their interaction; the prover wishes to maximize his expected reward; and his reward is maximized only if he the verifier correctly learns $f(x)$.

Rational proof systems are as powerful as their classical counterparts for polynomially many rounds of interaction, but are much more powerful when we only allow a constant number of rounds. Indeed, we prove that if $f \in \#P$, then f is computable by a one-round rational Merlin-Arthur game, where, on input x , Merlin’s single message actually consists of sending just the value $f(x)$. Further, we prove that CH, the counting hierarchy, coincides with the class of languages computable by a constant-round rational Merlin-Arthur game.

Our results rely on a basic and crucial connection between rational proof systems and *proper scoring rules*, a tool developed to elicit truthful information from experts.

1. INTRODUCTION

Many centuries have passed. Arthur continues to be bounded, probabilistic, and honest. Merlin, by contrast has changed a lot. Although remaining as intelligent as ever, he has become *rational* in the economic sense: that is, he now acts so as to maximize his own utility. Perhaps he can be excused. A true mathematician, for too long he had remained poor in a world in which money is increasingly valued. Eventually, poverty and frustration turned into disenchantment, and one day he no longer saw any point in enlarging his king’s knowledge for free. So, the last time Arthur asked him why he did not prove him another theorem, he answered with brutal honesty: ‘If you really you want me to prove a theorem to you, I want to be paid.’ After the initial shock, Arthur agreed to pay Merlin \$1 for any new theorem successfully proved. Proving promptly resumed.

This simple arrangement, however, did not last long. The king soon saw some new opportunities in moving from a chivalrous to a mercantilistic world. Namely, he saw how to use Merlin’s intelligence and rationality to obtain much faster proofs—that is, with fewer rounds of interaction—and proposed an alternative arrangement to the wizard.

Rational Arthur-Merlin Games. Focusing, more generally, on the evaluation of a given *function* (rather than a given *predicate*) f with domain D , they agreed as follows. On input $x \in D$, Arthur and Merlin keep turns exchanging messages as before, with Merlin going first and Arthur always responding with a random string. After a prescribed number of *rounds* (i.e., message pairs), a pre-specified, polynomial-time *reward* function R is evaluated on (x, \mathcal{T}) , where \mathcal{T} (the *transcript*) is the sequence of messages between the two players. Merlin gets paid $R(x, \mathcal{T})$. Since Merlin wants a positive amount of money and Arthur’s budget is limited, we require that $0 \leq R(x, \mathcal{T}) \leq c$ for some constant c .

Informally, the above arrangement constitutes a *rational Merlin-Arthur game* for f if, for all $x \in D$, Merlin maximizes his expected reward by revealing the true value of $f(x)$. Note that Arthur does not verify that the computation of $f(x)$ is correct. Instead, the correctness of $f(x)$ is guaranteed by Merlin’s unbounded computational power and his rationality.

Informally, $\text{RMA}[1]$ and $\text{RMA}[k]$ respectively denote the classes of functions f having a rational Merlin-Arthur game

with one round, and with k rounds, where k is a constant. We will work with both functional and decision versions of rational Merlin-Arthur games. To emphasize the distinction, we call the corresponding class of function problems $\text{FRMA}[k]$ and the corresponding class of decision problems $\text{DRMA}[k]$.

1.1 The Surprising Power of One-Round Rational Proofs

We prove that rational proof systems are amazingly powerful with just one round of interaction.

THEOREM 1. $\#P \subset \text{FRMA}[1]$.

One-round rational proofs continue to be powerful even when restricted to decision problems. In section 3, we define a decision analogue of $\text{FRMA}[1]$, which we call $\text{DRMA}[1]$ to emphasize that it is a class of decision problems. Recall that PP is the set of languages L for which there exists a non-deterministic Turing Machine M such that $x \in L$ if and only if more than half of the computational branches of M accept x . We can show that

THEOREM 2. $\text{PP} \subset \text{DRMA}[1]$

A corollary of Theorem 2 is that one-round rational M-A games are, under standard complexity assumptions, more powerful than their classical counterpart $\text{MA}[1]$

$\text{DRMA}[1]$ includes $\text{MA}[1]$. The inclusion is strict unless the polynomial hierarchy collapses.

Indeed, the inclusion is obvious since $\text{MA}[1] \subset \text{PP}$, and if it were not strict, then we would have $\text{coNP} \subset \text{PP} \subset \text{DRMA} \subset \text{MA}[1]$. But, unless the polynomial hierarchy collapses, $\text{coNP} \not\subset \text{MA}[1]$, which is a result due to [12].

1.2 A Characterization of Constant-Round Rational Proofs

In section 4 and appendix B we define a decision version of rational Merlin Arthur proofs over k rounds where k is a constant, which we call $\text{DRMA}[k]$. We show a complete characterization of $\text{DRMA} = \cup_{k \geq 0} \text{DRMA}[k]$ in terms of the *counting hierarchy*, a counting analogue of the polynomial hierarchy defined by Wagner [49].

THEOREM 3. $\text{DRMA} = \text{CH}$

This theorem has several implications. The first is a new characterization of CH in terms of interactive proofs. There are many open problems for CH , and we leave for future research whether this new characterization can solve them. The second is that, under reasonable complexity assumptions ($\text{CH} \subsetneq \text{PSPACE}$), rational proofs with constant rounds have limited power. Indeed, classical proofs with polynomially many rounds (IP), are strictly more powerful.

1.3 Two Other Results On Rational Proofs

Although easier and less surprising, the following results nonetheless are good to know as they give a more complete picture about rational proofs

- $\text{RIP} = \text{PSPACE}$. We give the proof of this theorem in appendix C
- $\text{RNP} = \text{NPO} \subset \text{FRMA}[1]$ and the inclusion is strict unless the polynomial hierarchy collapses.

Outline . The rest of the paper is divided as follows. Section 2 summarizes some related work. Section 3 and appendix A give proofs for theorems 1 and 2. Section 4 and appendix B give a proof of theorem 3. Section 6 gives a conclusion, discusses further related work and proposes future directions.

2. PRIOR AND RELATED WORK

2.1 Interactive Proofs and Delegation of Computation

The study of interactive proofs started with the work of Goldwasser, Micali and Rackoff [28] and Babai and Moran [6]. These papers were followed by a sequence of work [29, 26, 21, 9, 5, 23] which culminated with $\text{IP} = \text{PSPACE}$ [36, 45].

The classical analogue of our class RMA is the class MA [6]. There are two important differences between RMA and MA . The first is that MA is believed to be equal to NP . That is, randomness does not help the verifier, under standard complexity assumptions.¹ However, we show that $\text{NP} \subset \text{RMA}$ and the inclusion is strict unless the polynomial hierarchy collapses. The second key difference is that the MA hierarchy of proofs with constant number of rounds is known to collapse to the second level, that is, $\text{AM}[k] = \text{AM}[2]$ for constant k . We show that $\text{CP}_k \subset \text{RMA}[k] \subset \text{CP}_{k+1}$, so the RMA hierarchy does not collapse unless the counting hierarchy does as well. One problem that we leave open is whether one can improve the performance of RMA by allowing the verifier to have private coins. It is known that private coins do not help in classical Merlin Arthur protocols [29].

2.2 Rational Multiparty Computation

The contrast between classical interactive proofs and rational ones is that, in the first case, the prover can be either “Good” (submitting a correct proof) or “Malicious” (submitting an incorrect proof hoping to fool the prover). In the new setting, the prover can only fail in ways that maximize its utility.

The idea of replacing a potentially malicious agent with a rational one is not new. Indeed, it is central to the literature of *rational multiparty computation*. Shoham and Tennenholtz [46] study non-cooperative computation with a trusted party. In their model, there are multiple agents each of whom has one input to a function $y = f(x_1, \dots, x_n)$. Each agent prefers (1) learning the output y over not learning it and (2) given that they learn y , they prefer that as many

¹If $\text{PrBPP} = \text{PrP}$ then $\text{MA} = \text{NP}$ as shown in [32].

other people as possible *do not learn* y . Halpern and Teague [31] study the problem without a trusted party, showing that no mechanism running in fixed time can implement a rational multiparty computation. However, they show a randomized mechanism running which is expected to terminate and which can implement rational multiparty computation. Katz [34] gives a broad survey of rational computation and other interactions between game theory and cryptography.

2.3 Testing Experts

Sandroni [42] considers the problem of distinguishing between a knowledgeable expert, who knows some distribution \mathcal{D} , and an ignorant one. He shows that any test that is passed by the knowledgeable expert can also be passed (with some probability) by an ignorant person. Babaioff, Blumrosen, Lambert and Reingold [7] restrict the set of possible distributions that the expert may know, and show that when the information in \mathcal{D} is “valuable” then one can distinguish between knowledgeable experts and ignorant ones. Fortnow and Vohra [22] use complexity-theoretic tools to show an efficient test that *cannot be passed* by experts with limited computational power.

There are contrasts with our work that we want to highlight. The first is that in our work the expert always has infinite computational power (as opposed to the expert in [22]) and thus is always knowledgeable. The second is that the expert does not know a distribution \mathcal{D} , but rather the *deterministic* answer to some hard computational problem. The question of whether we can distinguish between a knowledgeable expert and an ignorant one (that is, one with limited computational power) remains open.

3. PROVING $\#P \subset \text{FRMA}[1]$

First we introduce some notation. Given a string $x \in \{0, 1\}^*$, $|x|$ represents the description length of x . We say a function $f : \{0, 1\}^* \rightarrow \{0, 1\}^*$ is *polynomially bounded* if $|f(x)| \leq p(|x|)$ for some polynomial p .

Given a polynomially bounded function f and an input x , we define a Rational Merlin-Arthur game with one-round. Informally, Arthur will ask Merlin for the value of $f(x)$. When Merlin answers y , he receives a reward $R(x, y)$. Ideally, Merlin should maximize his expected reward by answering $y = f(x)$. More formally, we can define one-round Rational Merlin Arthur games as follows.

DEFINITION 1 (RATIONAL MERLIN ARTHUR). *Let $f : \{0, 1\}^* \rightarrow \{0, 1\}^*$ be a polynomially bounded function computable in polynomial time. A **Rational Merlin Arthur game with one round** is defined by a randomized reward function $R : \{0, 1\}^* \times \{0, 1\}^* \rightarrow \mathbb{R}_{\geq 0}$ such that for all inputs $x \in \{0, 1\}^*$*

1. $R(x, y)$ is computable in time polynomial in $|x|, |y|$.
2. $\mathbb{E}[R(x, f(x))] > \mathbb{E}[R(x, y)]$ for any $y \neq f(x)$, where the expectation is taken over the coin tosses of R .

The class $\text{FRMA}[1]$ is defined as the set of all functions that have Rational Merlin Arthur Protocols with 1 round of communication. We now show the main theorem of this section

THEOREM 1. $\#P \subset \text{FRMA}[1]$

PROOF. We first show that $\#SAT \in \text{FRMA}[1]$. Consider the following probabilistic reward function R . For every formula $\phi = \phi(x_1, \dots, x_n)$ in conjunctive normal form, and every n -bit non-negative integer y , $R(\phi, y)$ is the value computed by the following process:

1. Select n uniformly random bits r_1, \dots, r_n and set $b = \phi(r_1, \dots, r_n)$.
2. Compute $\alpha(y) = (\frac{y}{2^n})^2 + (1 - \frac{y}{2^n})^2$.
3. If $b = 1$, then output $2\frac{y}{2^n} - \alpha(y) + 1$; else output $2(1 - \frac{y}{2^n}) - \alpha(y) + 1$.

Clearly R is computable in time polynomial in the first input by a probabilistic algorithm. Furthermore, one can show $R(x, y) \geq 0$. Thus, to prove that R is a one-round Merlin-Arthur game for $\#SAT$ we need to show that, for all conjunctive-normal-form formulas ϕ , letting $\# \phi$ be the number of satisfying assignments of ϕ , we have that

$$\operatorname{argmax}_y \mathbb{E}R(\phi, y) = \# \phi.$$

We could do this directly and in a self-contained manner, but by so doing we would hide all intuition and, worse, the important connection we wish to establish with strictly proper scoring rules. Let us instead adopt a better explanation.

In our procedure, Arthur computed two quantities: a random sample $b = \phi(r_1, \dots, r_n)$ and $\alpha(y) = (\frac{y}{2^n})^2 + (1 - \frac{y}{2^n})^2$. We now want to highlight the crucial connection between rational Merlin-Arthur games and scoring rules. Recall that Arthur wants to obtain $\# \phi$. This is an integer between 0 and 2^n , but it can also be interpreted as a random variable B over $\{0, 1\}$, giving the probability that a uniformly random selection of variables x_1, \dots, x_n will satisfy ϕ . Under this interpretation, b is a random sample from B . The fact that Merlin knows $\# \phi$ can be interpreted as saying that Merlin knows the distribution B . By sending the integer y , Merlin is de facto reporting a random variable Y over $\{0, 1\}$ such that $\Pr[Y = 1] = \frac{y}{2^n}$. Note that $Y = B$ if and only if $y = \# \phi$. Under this interpretation, Arthur wants to incentivize Merlin to report the true distribution B . The appropriate tool for solving this problem is a *proper scoring rule*. Since Arthur’s budget is fixed, we use Brier’s scoring rule because it is bounded in the interval $[-2, 0]$. However, using this rule would give Merlin a negative reward. This is why we use a linear transformation of Brier’s scoring rule (namely, we add 2), which does not affect incentives.

$$R(\phi, y) = \text{BSR}(Y, b) + 2 = \begin{cases} Y(1) - \|Y\|^2 + 1 & \text{if } b = 1 \\ Y(0) - \|Y\|^2 + 1 & \text{if } b = 0 \end{cases}$$

Recall that, for a general random variable Y , Brier’s scoring rule requires us to compute $\|Y\|^2 = \sum_{\omega \in \Omega} Y(\omega)^2$, which in general is hard to compute when $|\Omega|$ is large. In this case $|\Omega| = |\{0, 1\}| = 2$, and we have $\|Y\|^2 = (\frac{y}{2^n})^2 + (1 - \frac{y}{2^n})^2$,

which is the same as $\alpha(y)$ defined above. In other words, not only have we “reduced” rational proof systems to scoring rules, but to scoring rules over bit-distributions!

Because our protocol incentivizes Merlin with a proper scoring rule, he maximizes his expected reward when Y is the same as the distribution B from which the sample b is drawn. But b is a random variable which is 1 with probability $\frac{\# \phi}{2^n}$. Thus, Y matches B whenever $y = \# \phi$. This shows that $\#SAT \in \text{FRMA}[1]$.

Since any problem in $\#P$ is reducible to $\#SAT$ under one-to-one reductions, to complete the proof that $\#P \subset \text{FRMA}[1]$, we need to show that $\text{FRMA}[1]$ is closed under one-to-one reductions. Recall that a one-to-one reduction from a function f to another function g is a triple of polynomial time computable functions (α, β, γ) such that

1. For all x in the domain of f we have $y = f(x)$ if and only if $g(\alpha(x)) = \beta(y)$
2. For all x in the domain of f , let $w = \alpha(x)$. Then we have $g(w) = z$ if and only if $f(x) = \gamma(z)$.

Let $g \in \text{FRMA}[1]$ and let (α, β, γ) be a one-to-one reduction from some function f to g . We want to show that $f \in \text{FRMA}[1]$. Since $g \in \text{FRMA}[1]$, it is associated with a reward function $R_g(w, z)$ that incentivizes Merlin to compute the right answer to $g(w)$. Define a new reward function $R_f : \{0, 1\}^* \times \{0, 1\}^* \rightarrow \mathbb{R}$ such that for all x in the domain of f , $R_f(x, y) = S_g(w, z)$ where $w = \alpha(x)$, $z = \beta(y)$. Since S_g, α, β are polynomial time computable in $|x|$, R_f is also polynomial time computable in $|x|$.

Now we need to show that for all $y \neq f(x)$, we have $\mathbb{E}R_f(x, f(x)) > \mathbb{E}R_f(x, y)$. Let $w = \alpha(x)$, $z = \beta(y)$ as before. Note that since $y \neq f(x)$, and the reduction is one-to-one, we have $z = \beta(y) \neq \beta(f(x)) = g(w)$. With this fact in hand, we can write

$$\mathbb{E}R_f(x, f(x)) = \mathbb{E}R_g(w, \beta(f(x))) = \mathbb{E}R_g(w, g(w)) > \mathbb{E}R_g(w, z) = \mathbb{E}R_f(x, y)$$

Thus, we conclude that $\mathbb{E}R_f(x, f(x)) > \mathbb{E}R_f(x, y)$ for all $y \neq f(x)$. This shows that $f \in \text{FRMA}[1]$ and that this class is closed under one-to-one reductions.

To summarize, we have shown that $\#SAT \in \text{FRMA}[1]$ and that $\text{FRMA}[1]$ is closed under one-to-one reductions. Thus, $\#P \subset \text{FRMA}[1]$. *Q.E.D.*

3.1 Comparison with classical analogues

The classical analogue of $\text{FRMA}[1]$ is the complexity class MA of Merlin Arthur games. It should be intuitive that, since $\#P \subset \text{FRMA}[1]$, that rational Merlin Arthur games are much more powerful than classical ones. However, in order to make a proper comparison, we need to define a decision analogue of $\text{FRMA}[1]$.

DEFINITION 2. A language $L \in \{0, 1\}^*$ is in $\text{DRMA}[1]$ if and only if there exists a polynomial time predicate $\pi(x, y)$ and a reward function $R(x, y) \geq 0$ computable in randomized polynomial time such that

1. Given x , there exists a unique $y^*(x) = \text{argmax}_y \mathbb{E}[R(x, y)]$.
2. If $x \in L$, then $\pi(x, y^*(x)) = 1$.
3. If $x \notin L$, then $\pi(x, y^*(x)) = 0$.
4. If $y \neq y^*(x)$, $\pi(x, y)$ is arbitrary.

The predicate $\pi(x, y^*(x))$ simply helps Arthur distinguish whether y^* is a proof that $x \in L$ or $x \notin L$. We want to emphasize that y^* is *not* a proof and π is *not* a verifier. For example, for the language $\text{MAJSAT} \in \text{PP}$ (as we will show in appendix A) y^* will be the number of satisfying assignments to a given formula, and the predicate π will simply say if this number is greater or lower than $\frac{2^n}{2}$. Arthur cannot verify by himself that this answer is correct. The correctness is due to Merlin’s ability to solve hard problems, as well as Merlin following his own self-interest.

We can now state an analogue of theorem 1, which we prove in appendix A.

THEOREM 2. $\text{PP} \subset \text{DRMA}[1]$.

As mentioned and proved in the introduction, the fact that $\text{PP} \subset \text{DRMA}[1]$ implies the following corollary, which shows that – under widely believed complexity assumptions – classical Merlin Arthur protocols, introduced by [6], are strictly less powerful than Rational ones.

COROLLARY 1. $\text{MA} \subset \text{DRMA}[1]$ and the inclusion is strict unless the polynomial hierarchy collapses.

3.2 The crucial role of randomness

For classical interactive proofs, it is widely believed that $\text{P} = \text{RP}$ (randomness does not help). Indeed, if $\text{PrBPP} = \text{PrP}$ then $\text{MA} = \text{NP}$ [32]. We show this is not the case for the rational analogue of these classes.

First, we need to define RNP . We will define it as a function class. Informally, this is exactly the same definition as that of $\text{FRMA}[1]$ except that now the reward function $R(x, y)$ is deterministic. One can show that under this definition, RNP is the set NPO of NP optimization problems, defined as follows [4]:

DEFINITION 3. An NP optimization problem is given by a tuple $(\mathcal{I}, \mathcal{S}, R, \mathcal{L})$ where

- $\mathcal{I} \subset \{0, 1\}^*$ is a set of valid instances.
- For any instance $x \in \mathcal{I}$, $\mathcal{S}(x)$ is a set of feasible solutions of x .
- $R : \mathcal{I} \times \mathcal{S} \rightarrow \mathbb{R}$ is a function in FP that assigns a score to each (instance, solution) pair.
- $\mathcal{L} \subset \mathcal{I} \times \mathcal{S}$ is the set of (instance, solution) pairs (x, y^*) such that $y^* = \text{argmax}_y \mathbb{E}R(x, y)$.

NP and NPO are known to be equivalent under Turing reductions [4]. Since FRMA[1] contains #P, we have that, in contrast to the classical case, randomness is crucial for rational proofs.

COROLLARY 2. *RNP = NPO \subset FRMA[1] and the inclusion is strict unless the polynomial hierarchy collapses.*

4. RATIONAL INTERACTIVE PROOFS WITH MULTIPLE ROUNDS

When we consider Rational Interactive Proofs over multiple rounds, we can obtain a complete characterization of what these proofs can do. First, we have to define what multi-round rational proofs are. It will be more natural for our theorems to define Rational Interactive Proofs for decision problems. The definition for function problems is analogous.

DEFINITION 4 (INTERACTIVE PROTOCOLS). *A k -round interactive protocol consists of two randomized functions $P, V : \{0, 1\}^* \rightarrow \{0, 1\}^*$. Given an input x and a round i , we can define the transcript \mathcal{T}_i , the prover's view \mathcal{P}_i , and the verifier's view \mathcal{V}_i at round i as follows:*

- $\mathcal{T}_0, \mathcal{V}_0, \mathcal{W}_0 \stackrel{\text{def}}{=} \{x\}$.
- $a_i = P(\mathcal{P}_i)$, and $\mathcal{P}_i = (\mathcal{P}_{i-1}, \mathcal{T}_{i-1}, r_i)$ where r_i is the set of random coin tosses used to compute a_i , P 's message at round i .
- $b_i = V(\mathcal{V}_i)$, and $\mathcal{V}_i = (\mathcal{V}_{i-1}, \mathcal{T}_{i-1}, s_i, a_i)$ where s_i is the set of random coin tosses that the prover uses at time i and a_i is the prover's message at round i .
- $\mathcal{T}_i = (\mathcal{T}_{i-1}, a_i, b_i)$.

The protocol has public coins if r_i is included in the message a_i and s_i is included in the message b_i .

DEFINITION 5 (RATIONAL INTERACTIVE PROOF). *A language L has a Rational Interactive Proof if there exists an interactive protocol (P, V) , a reward function $R(\cdot, \cdot)$ and a polynomial-time predicate $\pi(\cdot, \cdot)$ such that, on input x*

- *The verifier's messages can be computed in time polynomial in $|x|$.*
- *All message lengths are polynomial in $|x|$.*
- *If the prover and verifier follow the protocol, the transcript \mathcal{T} is produced. For this transcript, we have $\pi(x, \mathcal{T}) = 1$ if $x \in L$ and $\pi(x, \mathcal{T}) = 0$ if $x \notin L$.*
- *The prover does not want to be the first to deviate from the protocol. If both prover and verifier follow the protocol correctly up to time i , and the prover's view is \mathcal{P}_i , then $P(\mathcal{P}_i) = \arg\max_{a_i} \mathbb{E}_{\mathcal{T}_k} [R(\mathcal{T}_k) | a_i, \mathcal{P}_i]$, where the expectation is taken over all possible complete transcripts \mathcal{T}_k that are compatible with the current prover's view \mathcal{P}_i . Furthermore $P(\mathcal{P}_i)$ is the unique input message that maximizes this expectation.*

DEFINITION 6 (DRMA, RIP). *Let k be a constant. The set of all languages which admit a k -round rational interactive proof with public coins is called DRMA[k]. We define $\text{DRMA} = \bigcup_{k: k > 0} \text{DRMA}[k]$.*

The set of all languages which admit a rational interactive proof with polynomially many rounds (with or without public coins) is called RIP.

We can now show the relative power of rational interactive proofs compared to classical ones. First, we show

THEOREM 3. *CH = DRMA.*

This not only completely determines FRMA, but also gives a new interactive proof characterization of the counting hierarchy, which might be of independent interest. We prove this theorem and recall definitions for the counting hierarchy in appendix B. One important aspect of our proof is that, when giving rational proofs for problems in CH, Merlin's rewards will always be given by scoring rules, or linear combinations of scoring rules.

When we use polynomially many rounds and possibly private coins, rational proofs are no more powerful than classical ones.

THEOREM 4. *RIP = IP = PSPACE.*

We give the proof of this theorem in appendix C

5. PROPER SCORING RULES FOR COMPUTATIONALLY BOUNDED AGENTS

We can think of scoring rules as a type of rational proof, where Arthur has access to a sample ω from some unknown distribution \mathcal{D} . Under this interpretation, we are interested in the case when Arthur can easily compute Merlin's reward. Both the Brier and Logarithmic rules can be interpreted as algorithms with black-box access to the distribution \mathcal{F} . On input x , the black-box will return $\mathcal{F}(x)$. One problem with Brier's scoring rule is that it seems to require $|\mathcal{F}|$ queries to the distribution, where $|\mathcal{F}|$ is the size of the domain Ω of \mathcal{F} . Good's scoring rule avoids this problem. To compute $\ln \mathcal{F}(\omega)$, one only needs to query \mathcal{F} on one input. Scoring rules with this property are called *local*. Thus, Brier's rule has the advantage of being bounded, but it is far from being local. Good's has the advantage of locality, but it is unbounded. It would be ideal to have a scoring rule that is both local and bounded. Unfortunately, this is not the case, as shown by the following lemma.

LEMMA 1 (McCARTHY - SAVAGE [38] [43]). *If $|\Omega| \geq 3$ then the only local scoring rules are of the form $S(\mathcal{F}, \omega) = a + b \ln \mathcal{D}(\omega)$ where $b > 0$.*

Given that local scoring rules cannot be bounded, the natural question is to ask whether there are *multi-local* rules that are bounded. Informally, we define a k -local scoring rule as one that only accesses its input distribution \mathcal{F} on k inputs. (A formal definition is given in appendix ??.) Are there k -local rules that are bounded? We show that, for $k \leq \frac{|\Omega|}{4}$, the answer is no.

THEOREM 5. *If $k \leq \frac{|\Omega|}{4}$, and S is a k -local scoring rule, then S is unbounded.*

All the scoring rules proposed so far interact with the input distribution \mathcal{F} by *querying* it on inputs $\omega_1, \dots, \omega_k$ to obtain the values $\mathcal{F}(\omega_1), \dots, \mathcal{F}(\omega_k)$. That is, classical scoring rules can be defined as algorithms A that can make calls to an evaluation oracle, which we call \mathcal{F}_E . That is, $A = A^{\mathcal{F}_E}$.

We note however, that querying is only one possible way of interacting with a distribution \mathcal{F} . Another natural way is *sampling* \mathcal{F} . That is, we find natural to consider a *sampling scoring rule* that is an algorithm A having access to two oracles: an evaluation oracle $\mathcal{F}_E(\cdot)$ and a sampling oracle \mathcal{F}_S which when given the empty string λ returns a random sample $x \leftarrow \mathcal{F}$. That is, a sampling scoring rule is an algorithm $A = A^{\mathcal{F}_E, \mathcal{F}_S}$. We can then prove

THEOREM 6. *There exists a sampling scoring rule $A^{\mathcal{F}_E, \mathcal{F}_S}$ that makes one call to \mathcal{F}_S and two calls to \mathcal{F}_E . Furthermore, the scoring rule is bounded and strictly proper.*

6. SUMMARY OF CONTRIBUTIONS AND FUTURE DIRECTIONS

Our contributions are a new type of proof system, a characterization of said proof system in terms of the counting hierarchy and new results on the complexity of scoring rules.

We see at least two interesting directions for future work. The first direction is in efficient delegation of computation. General interactive proofs are not practical and there has been plenty of work in reducing the number of messages exchanged as well as the amount of work that the verifier needs to do [39, 35, 33, 27, 14, 8, 40, 10, 24, 16, 17, 2, 11, 44, 18]. Delegation of computation requires the user to be able to *verify* that the computation is correct. We propose rational proofs as an alternative, where the correctness of the computation is guaranteed by economic reasons. Note that in our proof that $\#SAT \in \text{FRMA}[1]$, Arthur’s computation time is linear in the size of the input instance ϕ . Thus, Arthur is not only efficient but *linear*. This means that, for any problem with a linear time reduction to $\#SAT$, we can find a rational proof with only one round of communication and where the verifier acts in linear time. These are very efficient proofs for a very hard problem! An interesting question is to what extent we can provide rational proofs for other problems that are not $\#SAT$ where Arthur only needs to perform linear computations.

The second direction for future work is using our equivalence between DRMA and CH to solve open problems in counting. For example, one can show that a logarithmic space analogue of the counting hierarchy is equivalent to uniform TC^0 circuits [1]. Does the connection between rational proofs and the counting hierarchy still hold if we restrict the verifier to logarithmic space? Can this connection help us answer questions about circuit complexity? We leave these questions for future research.

As a final note, we want to remark that we can give other interpretations to rational proof systems outside economics.

The interpretation in this paper is that Merlin is self-interested and wants to maximize some “cash” reward. But this does not need to be the case for our complexity results to hold. One can think of the prover as some process that, given input x , outputs some y maximizing (or minimizing) some random function $R(x, y)$. Rationality gives one justification for this maximization process, but it does not need to be the only justification.

Acknowledgements. We thank Ronitt Rubinfeld, Alessandro Chiesa, Jing Chen and Shrenik Shah for helpful discussions.

APPENDIX

A. PROOF OF $PP \subset DRMA[1]$

Recall from section 3 the definition of $DRMA[1]$:

DEFINITION 7. A language $L \in \{0, 1\}^*$ is in $DRMA[1]$ if and only if there exists a polynomial time predicate $\pi(x, y)$ and a reward function $R(x, y) \geq 0$ computable in randomized polynomial time such that

1. Given x , there exists a unique $y^*(x) = \operatorname{argmax}_y \mathbb{E}[R(x, y)]$.
2. If $x \in L$, then $\pi(x, y^*(x)) = 1$.
3. If $x \notin L$, then $\pi(x, y^*(x)) = 0$.
4. If $y \neq y^*(x)$, $\pi(x, y)$ is arbitrary.

We now show that $PP \subset DRMA[1]$.

LEMMA 2. $PP \subset DRMA[1]$

PROOF. This follows from the proof that $\#P \subset FRMA[1]$. We showed that $\#SAT$ was contained in $FRMA[1]$ by constructing a function $R(\phi, y)$ that is maximized only when $y = \# \phi$. We use the same function to show that $MAJSAT$ is in $DRMA[1]$, which implies that $PP \subset DRMA[1]$.

Let ϕ be an input to $MAJSAT$. Let

$$\pi(\phi, y) = \begin{cases} 1 & \text{if } y \geq \frac{2^n}{2} \\ 0 & \text{otherwise} \end{cases}$$

Since $\operatorname{argmax}_y \mathbb{E}R(\phi, y) = \# \phi$, we have that if $\phi \in MAJSAT$, then $\pi(\phi, \operatorname{argmax}_y \mathbb{E}R(\phi, y)) = 1$, and if $\phi \notin MAJSAT$, then $\pi(\phi, \operatorname{argmax}_y \mathbb{E}R(\phi, y)) = 0$. This shows that $MAJSAT \in DRMA[1]$. To conclude that $PP \in DRMA[1]$, we need to show that $DRMA[1]$ is closed under one-to-one reductions. The proof is analogous to our proof that $FRMA[1]$ was closed under one-to-one reductions in section 3. *Q.E.D.*

We highlight again that the predicate $\pi(\phi, y)$ is efficiently computable. Contrast this with the very similar but believed to be hard predicate

$$\rho(\phi, y) = \begin{cases} 1 & \text{if } y = \# \phi \text{ and } y \geq \frac{2^n}{2} \\ 0 & \text{otherwise} \end{cases}$$

The key difference is that the predicate ρ verifies that $y = \# \phi$. The predicate π on the other hand trusts that Merlin is telling the truth (otherwise he wouldn't be maximizing his own utility), and simply decides whether the given y implies that $\phi \in MAJSAT$ or $\phi \notin MAJSAT$.

B. PROVING $DRMA = CH$

We prove in this appendix that $DRMA = CH$. Before we prove this, we recall the definition of the counting hierarchy, as well as a useful result, due to Toran [47]. We will need some details not in his paper, which is why we reprove this result here.

B.1 The Counting Hierarchy

The counting hierarchy CH was introduced by [49] as a counting analogue to the polynomial hierarchy. First, we recall the definition of this hierarchy. Our definitions follow the ones in [1]

DEFINITION 8 (COUNTING OPERATOR [1][49]). Let K be a class of languages. $C \cdot K$ is the class of all languages L for which there exists a polynomial p and a language $A \in K$ satisfying

$$x \in L \iff \#\{y : |y| \leq p(|x|) \text{ and } (x, y) \in A\} > \frac{1}{2} \cdot 2^{p(|x|)}$$

DEFINITION 9 (CH). The counting hierarchy is the union $\bigcup_{k:k>0} CP_k$ where the CP_k are defined recursively as follows

1. $CP_0 = P$
2. $CP_1 = PP$
3. $CP_k = C \cdot CP_{k-1}$

Toran [47] showed that $CP_k = PP^{CP_{k-1}}$ giving an alternative characterization to the counting hierarchy. We show one side of this result, as we will need this proof to show $CH \subset DRMA$.

LEMMA 3 (TORAN). $CP_k \subset PP^{CP_{k-1}}$ where the PP machine only makes one query to the CP_{k-1} oracle per computation branch.

PROOF. Let $L \in CP_k$. This means that there exists a polynomial p and a language $A \in CP_{k-1}$ such that

$$x \in L \iff \#\{y : |y| \leq p(|x|) \text{ and } (x, y) \in A\} > \frac{1}{2}.$$

Now consider a non-deterministic Turing Machine $M^A \in PP^A$ with oracle access to A . The non-deterministic machine does the following:

1. Non-deterministically choose y such that $|y| \leq p(|x|)$.
2. Ask whether $(x, y) \in A$. If $(x, y) \in A$, accept. Otherwise, reject.

Since $M^A \in PP^A$, it accepts if and only if more than half of its computation branches accept. But that means that M^A accepts if and only if $x \in L$. *Q.E.D.*

We will first prove that $CH \subset DRMA$

LEMMA 4. $CH \subset DRMA$

PROOF. We proceed by induction, showing that $CP_k \subset DRMA[k]$. Notice that multiple round protocols give rise

to a dynamic incentive problem. It is possible that Merlin will want to lie in the first query in order to get a bigger profit in queries 2, 3, ..., k . Merlin forsakes some profit today to make more profit tomorrow.

To avoid this incentive problem, we first assume that there are k independent Merlins, and we only ask one query to each one. Let k -DRMA be the class of languages which can be decided via rational Merlin Arthur games with k Merlins, where Arthur can only make one query to each Merlin. We will show that CP_k is reducible to k -DRMA. Then, to finish the proof, we will show that we can replace k Merlins with just one Merlin for this particular problem. The question of whether multiple Merlins are more powerful than one for rational proofs remains open.

We already showed the first step of this induction when we showed that $\text{PP} \subset \text{DRMA}[1]$. Note that $\text{DRMA}[1] = 1$ -DRMA.

Now assume that we know $\text{CP}_{k-1} \subset (k-1)$ -DRMA. We want to show $\text{CP}_k \subset k$ -DRMA. Recall that $\text{CP}_k = \text{PP}^{\text{CP}_{k-1}}$. Furthermore, we only need one query to the CP_{k-1} oracle per computation branch of the PP machine.

Consider a language $L \in \text{CP}_k$. Let $M^{\text{CP}_{k-1}}$ be a non-deterministic machine that accepts L if and only if more than half of its computation paths accept. We now exhibit a Rational Arthur-Merlin game with k Merlins to decide $x \in L$.

1. Arthur chooses a computation branch of machine M at random. He will need to make one query to the CP_{k-1} oracle. Since any language in CP_{k-1} can be decided by a rational MA proof with $k-1$ Merlins, Arthur simulates the query to CP_{k-1} by asking questions to $k-1$ Merlins.
2. Arthur reaches the end of his computation branch and writes down the output $b \in \{0, 1\}$.
3. Arthur brings in a new Merlin (the k^{th} one) and asks him how many paths of $M^{\text{CP}_{k-1}}$ accept. Merlin answers y .
4. Arthur creates the random variable $Y \in \{0, 1\}$ such that $\Pr[Y = 1] = \frac{y}{2^n}$. He rewards the k^{th} Merlin using Brier's scoring rule $\text{BSR}(Y, b) = Y(b) - \|Y\|^2 + 1$.²
5. If $y > 2^n/2$ then Arthur outputs $x \in L$. Otherwise, he outputs $x \notin L$.

By the inductive hypothesis, the first $(k-1)$ Merlins are being truthful, and their answers help Arthur simulate the query to CP_{k-1} . This gives Arthur a random output b from the machine $M^{\text{CP}_{k-1}}$. Since Merlin is rewarded using the scoring rule $\text{BSR}(Y, b)$, he will announce Y so it matches the distribution from which b is drawn. That is, he will announce y equal to the number of accepting computation branches in $M^{\text{CP}_{k-1}}$. Note that it does not matter if Arthur reveals

²Again, we shift BSR by 2 so the reward is always positive. This does not affect incentives.

his coin tosses (the random path he chooses in the machine $M^{\text{CP}_{k-1}}$).

This shows that any language in CP_k can be decided by a rational Merlin Arthur proof using k Merlins. Now all we need to do is show that we can simulate k Merlins using only one Merlin and k queries. We sketch the proof below.

Note that all of the interactions between Arthur and Merlin in the above proof are of the form:

1. Arthur samples b from some distribution B over $\{0, 1\}$.
2. Merlin announces some random variable $Y \in \{0, 1\}$ such that $\Pr[Y = 1] = \frac{y}{2^n}$ for some integer y .
3. Arthur rewards Merlin with $R(Y, b) = \text{BSR}(Y, b)$

Assume without loss of generality that we apply a linear transformation to BSR so that it always outputs a reward between $[0, 1]$. Let B be the distribution from which b is drawn (that is, the *true distribution*) and let Y be any distribution $Y \neq B$. If Merlin announces Y , he is *lying*. Define $\Delta = \min_{Y \neq B} \|\mathbb{E}_b[\text{BSR}(B, b) - \text{BSR}(Y, b)]\|$. We need the following two facts about Δ : (1) $\Delta > 0$, (2) Δ can be expressed in polynomially many bits.

Both facts are easy to show. Since BSR is a strictly proper scoring rule, we have $\Delta > 0$. Since we work with distributions Y such that $\Pr[Y = 1]$ can be expressed in polynomially many bits, we have Δ can also be expressed in polynomially many bits.

By definition, Δ is Merlin's minimum expected loss from deviating. Under a worst case scenario, Merlin could lie today if he expected to profit by more than Δ tomorrow and lose only Δ today.

Suppose on the other hand that Merlin *cannot* profit by more than Δ tomorrow. Then Merlin's incentive is to be truthful today. Let B be a truthful announcement for Merlin and let $Y \neq B$ be a deviation. If Merlin announces Y , he loses more than Δ units of utility today. His deviation cannot increase tomorrow's profit by more than Δ . Thus, in total Merlin loses utility by deviating today.

To incentivize Merlin to be truthful over k time periods, we reward him using the following scheme. Let x_1, \dots, x_k be Arthur's queries, and let b_1, \dots, b_k be the random samples he draws for each query. Let y_1, \dots, y_k be Merlin's answers and let Y_1, \dots, Y_k be the corresponding random variables in $\{0, 1\}^k$. Merlin's reward is given by

$$S(b_1, \dots, b_k, Y_1, \dots, Y_k) = \sum_{i=1}^k \gamma^{i-1} \text{BSR}(Y_i, b_i),$$

where $\gamma = \frac{\Delta}{1+\Delta}$ is chosen so that $\sum_{i=2}^k \gamma^{i-1} < \frac{\gamma}{1-\gamma} \leq \Delta$.³

³This technique is inspired by the discounted market scoring rules of [15], who used it to argue approximate truthfulness for scoring rules in dynamic environments. Because of the special nature of our distributions, we can obtain exact truthfulness.

This choice guarantees that, at any time i , Merlin's loss from deviating from the truth is strictly larger than any gain he would obtain from his deviation in future periods. Thus, Merlin always wants to be truthful.

Q.E.D.

We have shown that $\text{CH} \subset \text{DRMA}$. We now show the opposite direction. We will use the following definition and facts in the proof.

DEFINITION 10. An NP optimization problem **with access to an oracle A** is given by a tuple $(\mathcal{I}, \mathcal{S}, R, \mathcal{L})$ where

1. $\mathcal{I} \subset \{0, 1\}^*$ is a set of valid instances.
2. For any instance $x \in \mathcal{I}$, $\mathcal{S}(x)$ is a set of feasible solutions of x .
3. $R : \mathcal{I} \times \mathcal{S} \rightarrow \mathbb{R}$ is a function in FP^A that assigns a score to each (instance, solution) pair and has **oracle access to A** .
4. $\mathcal{L} \subset \mathcal{I} \times \mathcal{S}$ is the set of (instance, solution) pairs (x, y^*) such that $y^* = \arg\max_y \mathbb{E}R(x, y)$.

LEMMA 5. The following facts are true

1. The set of NPO^A of NP optimization problems with oracle access to an arbitrary language A is Turing reducible to the set NP^A .
2. For any oracle A , $\text{NP}^A \subset \text{PP}^A$.
3. $\text{PP}^{\#P}$ is Turing equivalent to PP^{PP} .
4. If $f(x; r)$ is a randomized function computable in polynomial time that depends on random coins r , then $\mathbb{E}_r[f(x; r)]$ can be computed in $\text{P}^{\#P}$.

PROOF PROOF SKETCH. We sketch justifications for these facts.

1. It is known that NPO is Turing reducible to NP. Given an optimization problem \mathcal{P} in NPO that asks to find y maximizing some function $R(x, y)$, one can create an analogous decision problem \mathcal{P}_D that asks, given x and R and k , whether there exists y such that $R(x, y) \geq k$. With some modifications to break ties, one can show a Turing reduction that uses answers to \mathcal{P}_D and binary search over the parameter k to solve the optimization problem $R(x, y)$ (see theorem 1.5 in [4]). The reduction goes through even if we allow both the optimization and decision problems to have access to some oracle A . Whenever the optimization problem would need to query A , the reduction to the decision problem can also query A .
2. It is known that $\text{NP} \subset \text{PP}$. Furthermore, the machine model in both cases is the same: non-deterministic Turing machines. Thus $\text{NP}^A \subset \text{PP}^A$.

3. Given an oracle that answers queries to PP, we can, in polynomial time, answer problems in $\#P$ using binary search. Since machines in PP can implement binary search, $\text{PP}^{\#P} = \text{PP}^{\text{PP}}$.

4. We want to show that if $f(x; r)$ is computable in polynomial time then $\mathbb{E}_r f(x; r)$ is computable in $\text{P}^{\#P}$. Let $y_k \cdot 2^k + \dots + y_1 \cdot 2 + y_0 = f(x)$ be the output of $f(x)$ written in binary. The output bits y_k, \dots, y_0 are random variables that depend on the sequence of coin tosses r . By linearity of expectation $\mathbb{E}_r[\sum_{i=0}^k y_i 2^i] = \sum_{i=0}^k \mathbb{E}_r[y_i] \cdot 2^i$. Thus, it suffices to compute the expectation of each bit. Let $M_i(x)$ be a probabilistic polynomial time algorithm that tosses $\text{poly}(|x|)$ coins and computes the i^{th} bit of $y = f(x)$. The expected value of y_i is equal to the number of accepting computation paths of M_i , divided by the total number of paths, which is $2^{\text{poly}(|x|)}$. The number of accepting paths can be computed in $\#P$. Thus, we can compute $\mathbb{E}[y_i]$ for each bit using an oracle to $\#P$, and we can reconstruct $\sum_{i=0}^k 2^i \mathbb{E}y_i = \mathbb{E}y = \mathbb{E}f(x)$ in polynomial time

Q.E.D.

Now we prove that $\text{DRMA} \subset \text{CH}$.

LEMMA 6. $\text{DRMA} \subset \text{CH}$

PROOF. We proceed by first showing that $\text{DRMA}[1] \subset \text{PP}^{\#P} = \text{PP}^{\text{PP}} = \text{CP}_2$. Our proof then builds on this fact inductively.

Let L be a language in $\text{DRMA}[1]$. By definition, there exists a randomized function $R(x, y)$ and a predicate $\pi(x, y)$ such that

1. Given x , there is a unique $y^* = \arg\max_y \mathbb{E}_r[R(x, y; r)]$ where r are the random coin tosses of R .
2. $\pi(x, y^*) = 1$ if $x \in L$ and $\pi(x, y^*) = 0$ if $x \notin L$.

Now consider the following optimization problem. Given input $x \in \{0, 1\}^*$, we want to find y^* such that $y^* = \arg\max_y \mathbb{E}_r R(x, y; r)$ where r are the random coin tosses of R . This problem is in $\text{NPO}^{\#P}$ because for every x, y there exists an algorithm in $\text{P}^{\#P}$ that computes $E(x, y) =_{\text{def}} \mathbb{E}_r R(x, y; r)$. Such an algorithm exists as shown in lemma 5. It is a problem in $\text{NPO}^{\#P}$ because we want to find y^* maximizing $E(x, y)$ and $E(x, y)$ is in $\text{FP}^{\#P}$.

Since $\text{NPO}^{\#P}$ is Turing reducible to $\text{NP}^{\#P}$ (again, by lemma 5), there exists some non-deterministic machine $M^{\#P}$ such that $M^{\#P}(x)$ accepts if and only if it finds y^* maximizing $E(x, y)$.

A $\text{PP}^{\#P}$ machine can simulate the solution to this optimization problem and look at y^* . Then it evaluates $\pi(x, y^*)$. If $\pi(x, y^*) = 1$ then it outputs $x \in L$. Otherwise it outputs

$x \notin L$. This shows that $L \in \text{PP}^{\#P} = \text{PP}^{\text{PP}} = \text{CP}_2$. Thus, $\text{DRMA}[1] \subset \text{CP}_2$.

We now proceed by induction. We want to show that $\text{DRMA}[k]$ is contained in some level of the counting hierarchy. Assume that we have already shown that $\text{DRMA}[k-1]$ is in CP_k . Let L be a language in $\text{DRMA}[k]$. This means that there is a k -round rational Arthur Merlin game for deciding $x \in L$ or $x \notin L$.

Let (P, V) be an interactive k -round rational game for deciding $x \in L$. Define a language K_{yes} such that $(a_1, b_1, x) \in K_{yes}$ if and only if

1. $a_1 = P(x; r_1), b_1 = V(x, a_1; s_1)$ are correct first messages for the interactive protocol (P, V) .
2. The final transcript \mathcal{T} of a protocol that starts with messages a_1, b_1 satisfies $\pi(x, \mathcal{T}) = 1$.

Similarly, we can define a language K_{no} such that $(a_1, b_1, x) \in K_{no}$ if and only if a_1, b_1 are correct first messages and $\pi(x, \mathcal{T}) = 0$.

Informally, (K_{yes}, K_{no}) define a *promise problem* that can be solved via a rational Merlin Arthur game with $k-1$ rounds. We have not defined what it means for a Rational Merlin Arthur game to solve a promise problem. We will resolve this technicality after we define the protocol for (K_{yes}, K_{no}) . The game proceeds as follows: on input (a_1, b_1, x)

1. The prover defines his initial view to be $\mathcal{P}_2 = (a_1, b_1, x)$. He outputs $a_2 = P(\mathcal{P}_2)$.⁴
2. The verifier defines his initial view to be $\mathcal{V}_2 = (a_1, b_1, x, a_2)$. He outputs $b_2 = V(\mathcal{V}_2)$.
3. The protocol continues as in the original protocol for (P, V) .

In order to resolve the technicality that (K_{yes}, K_{no}) is a promise problem and not a decision problem, we extend the protocol to handle inputs (a, b, x) where $(a, b, x) \notin K_{yes} \cup K_{no}$. Since the protocol uses public coins, Arthur can check that $a = P(x), b = V(x, a)$. If this is the case, then (a, b, x) is either in K_{yes} or K_{no} . If it isn't the case, Arthur aborts the protocol and gives Merlin a reward $R(x, \mathcal{T}) \equiv 0$. It is clear that Arthur can use this mechanism to decide whether $x \in K_{yes}$ or $x \notin K_{yes}$ (either by showing $x \in K_{no}$ or by aborting the protocol). Thus, $K_{yes} \in \text{DRMA}[k-1]$.

Since this game has $k-1$ rounds, and we assumed $\text{DRMA}[k-1] \subset \text{CP}_k$, we have $K_{yes} \in \text{CP}_k$. Consider now the following non-deterministic Turing Machine $M^{K_{yes}}$ in $\text{PP}^{K_{yes}, \#P}$ that decides whether $x \in L$. Given input x , M does the following

⁴Here we use the fact that Rational Merlin Arthur games use public coins. This means that the random coins r_1, s_1 used by prover and verifier are sent as part of the messages a_1, b_1 . If they were not sent as part of these messages, then we cannot simulate the prover's view on input (a_1, b_1, x) because we don't have the prover's coin tosses.

1. Non-deterministically choose a first message a_1 .
2. Compute $E(x, a_1) = \mathbb{E}_{r; a_2, b_2, \dots, a_k, b_k} [R(x, a_1, b_1, \dots, a_k, b_k)]$. We can do this in $\text{P}^{\#P} \subset \text{PP}^{\#P}$ according to lemma 5.
3. Figure out a_1^* that maximizes $E(x, a_1)$. This is an algorithm in $\text{NPO}^{\#P}$, which can be simulated by our $\text{PP}^{\#P}$ machine.
4. Compute $b_1^* = V(a_1^*)$
5. Given (x, a_1^*, b_1^*) , ask the oracle whether it is in K_{yes} or K_{no} . Note that a_1^*, b_1^* are valid starting messages for input x , so if $x \notin K_{yes}$ it must be the case that $x \in K_{no}$.
6. Output $x \in L$ if $(x, a_1^*, b_1^*) \in K_{yes}$, and output $x \notin L$ otherwise.

This algorithm is a non-deterministic algorithm with oracle access to K_{yes} and $\#P$ which decides L . Since K_{yes} is in CP_k and we can simulate queries to $\#P$ using a $\text{PP} \subset \text{CP}_k$ oracle, we conclude $L \in \text{PP}^{\text{CP}_k} = \text{CP}_{k+1}$. Since $L \in \text{DRMA}[k]$, we conclude $\text{DRMA}[k] \subset \text{CP}_{k+1}$. This proves the inductive step, which shows that $\text{DRMA} \subset \text{CH}$.

Q.E.D.

C. PROVING $\text{RIP} = \text{PSPACE}$

We have shown that, for constant number of rounds, rational proofs are strictly more powerful than classical ones. We now show that this is not the case if we allow polynomially many rounds of interaction.

THEOREM 7. $\text{RIP} = \text{PSPACE}$

PROOF. We first show $\text{RIP} \subset \text{PSPACE}$. Let $L \in \text{RIP}$. By the definition above, there exists a protocol given by algorithms (P, V) and a randomized reward function R such that the prover never maximizes his expected reward by being the first to deviate from the protocol given by (P, V) .

We now show that L is a language that can be decided in polynomial space. Consider a Turing Machine that on input x performs the following steps

1. Keep state variables $\mathcal{P}, \mathcal{V}, \mathcal{T}$ where \mathcal{P} will simulate the prover's view, \mathcal{V} will simulate the verifier's view and \mathcal{T} will simulate the transcript. Initialize $\mathcal{P}_0 = \mathcal{V}_0 = \{x\}$.
2. For $i = 1 \dots k = \text{poly}(|x|)$
 - (a) Update the prover's view $\mathcal{P}_i = (\mathcal{P}_{i-1}, \mathcal{T}_{i-1}, r_i)$ where r_i is a sequence of random coins.
 - (b) Given that the protocol has been followed so far, the prover will choose a_i that maximizes $\mathbb{E}_{\mathcal{T}} [R(x, \mathcal{T}) | \mathcal{P}_i, a_i]$. This requires the computation of $R(x, \mathcal{T})$ over all possible final transcripts \mathcal{T} that are compatible with the prover's current view \mathcal{P}_i . This can be done in polynomial space, since each transcript only takes polynomial space. Furthermore R only

uses polynomially many random coins, so its expectation when we fix a \mathcal{T} can also be computed in polynomial space.

- (c) Update the verifier's view $\mathcal{V}_i = (\mathcal{V}_{i-1}, \mathcal{T}_{i-1}, a_i, s_i)$ where s_i is a sequence of random coins.
- (d) Compute the verifier's message $b_i = V(\mathcal{V}_i)$. This can be computed in polynomial time, and thus, in polynomial space.
- (e) Update $\mathcal{T}_i = (\mathcal{T}_{i-1}, a_i, b_i)$.

The above process simulates the Rational Interactive Proof in polynomial space. It finishes by outputting a transcript \mathcal{T} such that $\pi(x, \mathcal{T}) = 1$ if $x \in L$ and $\pi(x, \mathcal{T}) = 0$ if $x \notin L$. Since π can be computed in polynomial time, it can also be computed in polynomial space. Thus, we presented a PSPACE machine that decides L .

Q.E.D.

This inclusion is tight:

LEMMA 7. $PSPACE \subset RIP$.

PROOF. Since $IP = PSPACE$ [45], it suffices to show $IP \subset RIP$. Indeed, let L be a language that has an interactive proof. Consider the following rational interactive proof for L . Given an input x

- 1. If $x \in L$, prover and verifier compute an interactive proof π that $x \in L$. The prover's reward is $R(x, \pi) = 1$ if the verifier accepts the proof and zero otherwise.
- 2. If $x \notin L$, prover and verifier compute an interactive proof π that $x \notin L$. The prover's reward is $R(x, \pi) = 1$ if the verifier accepts the proof and zero otherwise.

In both cases, the prover maximizes his expected reward by giving a correct proof. Using this protocol, a verifier can decide whether $x \in L$ or $x \notin L$.⁵ Thus, L admits a rational interactive proof with polynomially many rounds.

Q.E.D.

4. REFERENCES

- [1] Eric W. Allender and Klaus W. Wagner. Counting hierarchies: Polynomial time and constant depth circuits, 1990.
- [2] B. Applebaum, Y. Ishai, and E. Kushilevitz. Computationally private randomizing polynomials and their applications. *Computational Complexity*, 15(2):115–162, 2006.
- [3] S. Arora and B. Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009.
- [4] Crescenzi P. Gambosi G. Kann V. Marchetti-Spaccamela A. Ausiello, G. and M. Protasi. *Complexity and approximation: combinatorial optimization problems and their approximability properties*. Springer Verlag, 1999.
- [5] L. Babai, L. Fortnow, and C. Lund. Non-deterministic exponential time has two-prover interactive protocols. *Computational complexity*, 1(1):3–40, 1991.
- [6] L. Babai and S. Moran. Arthur-merlin games: a randomized proof system, and a hierarchy of complexity classes. *Journal of Computer and System Sciences*, 36(2):254–276, 1988.
- [7] M. Babaioff, L. Blumrosen, N.S. Lambert, and O. Reingold. Only valuable experts can be valued. In *Proceedings of the 12th ACM conference on Electronic commerce*, pages 221–222. ACM, 2011.
- [8] B. Barak and O. Goldreich. Universal arguments and their applications. *ccc*, page 0194, 2002.
- [9] M. Ben-Or, S. Goldwasser, J. Kilian, and A. Wigderson. Multi-prover interactive proofs: How to remove intractability assumptions. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 113–131. ACM, 1988.
- [10] S. Benabbas, R. Gennaro, and Y. Vahlis. Verifiable delegation of computation over large datasets. 2011.
- [11] Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. 3rd Innovations in Theoretical Computer Science, 2012.
- [12] R.B. Boppana and J. Hastad Stathis. Does-np have short interactive proofs? *Information Processing Letters*, 25(2):127–132, 1987.
- [13] G.W. Brier. Verification of forecasts expressed in terms of probability. *Monthly weather review*, 1950.
- [14] R. Canetti, B. Riva, and G.N. Rothblum. Practical delegation of computation using multiple servers. In *Proceedings of the 18th ACM conference on Computer and communications security*, pages 445–454. ACM, 2011.
- [15] Y. Chen, S. Dimitrov, R. Sami, D.M. Reeves, D.M. Pennock, R.D. Hanson, L. Fortnow, and R. Gonen. Gaming prediction markets: Equilibrium strategies with a market maker. *Algorithmica*, 58(4):930–969, 2009.
- [16] K.M. Chung, Y. Kalai, and S. Vadhan. Improved delegation of computation using fully homomorphic encryption. *Advances in Cryptology-CRYPTO 2010*, pages 483–501, 2010.
- [17] G. Cormode, J. Thaler, and K. Yi. Verifying computations with streaming interactive proofs.
- [18] Ivan Damgård, Sebastian Faust, and Carmit Hazay. Secure two-party computation with low communication. Cryptology ePrint Archive, Report 2011/508, 2011.
- [19] A.P. Dawid, S. Lauritzen, and M. Parry. Proper local scoring rules on discrete sample spaces. *Arxiv preprint arXiv:1104.2224*, 2011.
- [20] W. Ehm and T. Gneiting. Local proper scoring rules of order two. *Arxiv preprint arXiv:1102.5031*, 2011.

⁵It does not matter if the proof fails with some very low probability. Arthur trusts Merlin to always give a correct answer and correct proof, because Merlin maximizes his utility this way.

- [21] L. Fortnow, J. Rompel, and M. Sipser. On the power of multi-prover interactive protocols. *Theoretical Computer Science*, 134(2):545–557, 1994.
- [22] L. Fortnow and R.V. Vohra. The complexity of forecast testing. *Econometrica*, 77(1):93–105, 2009.
- [23] Lance Fortnow and Michael Sipser. Are there interactive protocols for co-np languages? *Information Processing Letters*, 28:249–251, 1988.
- [24] R. Gennaro, C. Gentry, and B. Parno. Non-interactive verifiable computing: Outsourcing computation to untrusted workers. *Advances in Cryptology-CRYPTO 2010*, pages 465–482, 2010.
- [25] T. Gneiting and A.E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007.
- [26] O. Goldreich, S. Micali, and A. Wigderson. Proofs that yield nothing but their validity and a methodology of cryptographic protocol design. In *Foundations of Computer Science, 1986., 27th Annual Symposium on*, pages 174–187. IEEE, 1988.
- [27] S. Goldwasser, Y.T. Kalai, and G.N. Rothblum. Delegating computation: interactive proofs for muggles. In *Proceedings of the 40th annual ACM symposium on Theory of computing*, pages 113–122. ACM, 2008.
- [28] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof-systems. In *Proceedings of the seventeenth annual ACM symposium on Theory of computing*, pages 291–304. ACM, 1985.
- [29] S. Goldwasser and M. Sipser. Private coins versus public coins in interactive proof systems. In *Proceedings of the eighteenth annual ACM symposium on Theory of computing*, pages 59–68. ACM, 1986.
- [30] I.J. Good. Rational decisions. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 107–114, 1952.
- [31] J. Halpern and V. Teague. Rational secret sharing and multiparty computation. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 623–632. ACM, 2004.
- [32] R. Impagliazzo and A. Wigderson. $P = BPP$ if e requires exponential circuits: Derandomizing the xor lemma. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 220–229. ACM, 1997.
- [33] Y. Kalai and R. Raz. Probabilistically checkable arguments. *Advances in Cryptology-CRYPTO 2009*, pages 143–159, 2009.
- [34] J. Katz. Bridging game theory and cryptography: Recent results and future directions. *Theory of Cryptography*, pages 251–272, 2008.
- [35] J. Kilian. A note on efficient zero-knowledge proofs and arguments. In *Proceedings of the twenty-fourth annual ACM symposium on Theory of computing*, pages 723–732. ACM, 1992.
- [36] C. Lund, L. Fortnow, H. Karloff, and N. Nisan. Algebraic methods for interactive proof systems. *Journal of the ACM (JACM)*, 39(4):859–868, 1992.
- [37] W. Mantel. Problem 28. *Wiskundige Opgaven*, 10:60–61, 1906.
- [38] J. McCarthy. Measures of the value of information. *Proceedings of the National Academy of Sciences of the United States of America*, 42(9):654, 1956.
- [39] S. Micali. Computationally sound proofs. *SIAM Journal on Computing*, 30(4):1253–1298, 2001.
- [40] C. Papamanthou, R. Tamassia, and N. Triandopoulos. Optimal verification of operations on dynamic sets. In *Proc. International Cryptology Conference (CRYPTO)*, 2011.
- [41] M. Parry, A.P. Dawid, and S. Lauritzen. Proper local scoring rules. *Arxiv preprint arXiv:1101.5011*, 2011.
- [42] A. Sandroni. The reproducible properties of correct forecasts. *International Journal of Game Theory*, 32(1):151–159, 2003.
- [43] L.J. Savage. Elicitation of personal probabilities and expectations. *Journal of the American Statistical Association*, pages 783–801, 1971.
- [44] Aviad Rubinfeld Shafi Goldwasser, Huijia Lin. Delegation of computation without rejection problem from designated verifier cs-proofs. Cryptology ePrint Archive, Report 2011/456, 2011.
- [45] A. Shamir. $IP = PSPACE$. *Journal of the ACM (JACM)*, 39(4):869–877, 1992.
- [46] Y. Shoham and M. Tennenholtz. Non-cooperative computation: Boolean functions with correctness and exclusivity. *Theoretical Computer Science*, 343(1-2):97–113, 2005.
- [47] J. Torán. Complexity classes defined by counting quantifiers. *Journal of the ACM (JACM)*, 38(3):752–773, 1991.
- [48] P. Turán. On an extremal problem in graph theory (in hungarian), mat. *Fiz. Lapok*, 48:436–452, 1941.
- [49] K.W. Wagner. The complexity of combinatorial problems with succinct input representation. *Acta Informatica*, 23(3):325–356, 1986.