Rational Proofs

Pablo Daniel Azar
MIT, CSAIL
32 Vassar Street
Cambridge, MA, 02139
azar@csail.mit.edu

Silvio Micali*
MIT, CSAIL
32 Vassar Street
Cambridge, MA, 02139
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.

Categories and Subject Descriptors

F.1.3 [**Theory of Computation**]: Complexity Measures and Classes

General Terms

Economics, Theory

Keywords

Interactive Proofs, Rational Cryptography, Counting Hierarchy

*Partially funded by the Office of Naval Research, award number N00014-09-1-0597

STOC12, May 19–22, 2012, New York, New York, USA.

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, RMA[1] and 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 distinc-

tion, we call the corresponding class of function problems FRMA[k] and the corresponding class of decision problems 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. *$\#P \subset FRMA[1]$.*

One-round rational proofs continue to be powerful even when restricted to decision problems. In section 3, we define a decision analogue of FRMA[1], which we call 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 nondeterministic 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. *$PP \subset DRMA[1]$*

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

DRMA[1] includes MA[1]. The inclusion is strict unless the polynomial hierarchy collapses.

Indeed, the inclusion is obvious since MA[1] $\subset$ PP, and if it were not strict, then we would have coNP $\subset$ PP $\subset$ DRMA $\subset$ MA[1]. But, unless the polynomial hierarchy collapses, coNP $\not\subset$ MA[1], which is a result due to [11].

1.2 A Characterization of Constant-Round Rational Proofs

In section 4 and section 5 we define a decision version of rational Merlin Arthur proofs over k rounds where k is a constant, which we call DRMA[k]. We show a complete characterization of DRMA $= \cup_{k:k>0}$DRMA[k] in terms of the *counting hierarchy*, a counting analogue of the polynomial hierarchy defined by Wagner [43].

THEOREM 3. *$DRMA = CH$*

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 (CH $\subsetneq$ 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

- RIP = PSPACE. We give the proof of this theorem in section 6
- RNP = NPO $\subset$ FRMA[1] and the inclusion is strict unless the polynomial hierarchy collapses.

1.4 Using Scoring Rules

The underlying theme behind interactive proofs is *information asymmetry*. Merlin knows the answer to some hard problem, and Arthur does not. An alternative case of information asymmetry is when there is uncertainty about the world. In this case, Merlin knows a distribution $\mathcal{D}$ over the possible states of the world and Arthur does not. In this scenario, it is well known how to incentivize Merlin to reveal his knowledge: Arthur should use a *strictly proper scoring rule*. Indeed, we show that strictly proper scoring rules and rational interactive proofs are tightly connected.

We prove Theorems 1, 2 and 3 by using strictly proper scoring rules. These incentivize an expert to reveal a distribution $\mathcal{D}$ over a set Ω. Let there be two parties: Merlin, who knows $\mathcal{D}$, and *Arthur*, who has no information about $\mathcal{D}$ and wishes Merlin to tell him (an encoding of) $\mathcal{D}$. A *strictly proper scoring rule*[1] is a function S such that, for all distributions $\mathcal{D}'$ different from $\mathcal{D}$:

$$\sum_{\omega\in\Omega} \mathcal{D}(\omega)S(\mathcal{D},\omega) > \sum_{\omega\in\Omega} \mathcal{D}(\omega)S(\mathcal{D}',\omega).$$

Assume now that, after Merlin reports a distribution $\mathcal{X}$, a sample from $\mathcal{D}$ becomes available and that Arthur rewards Merlin with the amount $S(\mathcal{X},\omega)$. Then a rational Merlin wants to announce the true distribution $\mathcal{D}$. That is, it becomes *dominant* for Merlin to report $\mathcal{D}$ rather then another distribution $\mathcal{D}'$.

Many strictly proper scoring rules are known by now[2]: two popular ones [28, 12] are,

Good's rule, $S(\mathcal{D},\omega) = \log(\mathcal{D}(\omega))$, and
Brier's rule, $S(\mathcal{D},\omega) = 2\mathcal{D}(\omega) - \sum_{\omega\in\Omega} \mathcal{D}(\omega)^2 - 1$.

Outline .

The rest of the paper is divided as follows. Section 2 summarizes some related work. We give proofs of theorems 1 and 2 in section 3. Section 4 gives a definition of Rational Merlin Arthur games with multiple rounds, and section 5 give a proof of theorem 3. Section 6 gives a proof that $RIP = PSPACE$. We present independent new results on scoring rules in section 7. Section 8 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 [26] and Babai and Moran [5]. These papers were followed by a sequence of work [27, 24, 19, 8, 4, 21] which culminated with IP = PSPACE [34, 39].

The classical analogue of our class RMA is the class MA [5]. 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.[3] However, we show

[1]For brevity and when the context is clear, we will often refer to strictly proper scoring rules simply as *proper scoring rules* or *scoring rules*.

[2]The interested reader is referred to a paper by Gneiting and Rafterty [23], which includes a comprehensive survey.

[3]If PrBPP $= Pr$P then MA = NP as shown in [30].

that NP $\subset$ 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,a $AM[k] = AM[2]$ for constant k. We show that $\mathrm{CP}_k \subset \mathrm{RMA}[k] \subset \mathrm{CP}_{2k+1}$, 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 [27].

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 [40] study non-cooperative computation with a trusted party. In their model, there are multiple agents each of whom has one input to a function $y = f(x_1, ..., x_n)$. Each agent prefers (1) learning the output y over not learning it and (2) given that they learn y, they prefer that as many other people as possible *do not learn* y. Halpern and Teague [29] 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 [32] gives a broad survey of rational computation and other interactions between game theory and cryptography.

2.3 Testing Experts

Sandroni [37] considers the problem of distinguishing between a knowledgable expert, who knows some distribution $\mathcal{D}$, and an ignorant one. He shows that any test that is passed by the knowledgable expert can also be passed (with some probability) by an ignorant person. Babaioff, Blumrosen, Lambert and Reingold [6] 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 knowledgable experts and ignorant ones. Fortnow and Vohra [20] 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 [20]) and thus is always knowledgable. 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 knowledgable expert and an ignorant one (that is, one with limited computational power) remains open.

2.4 Refereed Games and multiple provers

Feige and Kilian [18] propose the complexity class $RG(k)$ of refereed games with k rounds. This class can be interpreted as the set of languages L such that a polynomial time referee can decide whether a string x is in L by setting up a game between two infinitely powerful experts. If the first expert wins the game with non-negligible advantage, then $x \in L$. If the second expert wins the game with non-negligible advantage, then $x \notin L$. Feige and Killian show that if the experts have one round of communication, this procedure can decide any language in $PSPACE$. If they have polynomially many rounds of communication, then this procedure can decide any language in $NEXP$.

Refereed games play the experts against each other, using them to verify that each expert's computation is correct. In contrast, we only have one prover, and cannot verify (either by ourselves, or by using another expert) that the prover is giving us correct answers to problems in $\#P$. Instead, we rely on incentives to trust the correctness of the prover's answer.

3. PROVING #P $\subset$ 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 $f : \{0,1\}^* \to \{0,1\}^*$ 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 $f : \{0,1\}^* \to \{0,1\}^*$ be a polynomially bounded function. A **Rational Merlin Arthur game with one round** is defined by a randomized reward function $R : \{0,1\}^* \times \{0,1\}^* \to \mathbb{R}_{\geq 0}$ 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 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. *$\#P \subset FRMA[1]$*

PROOF. We first show that $\#SAT \in$ FRMA[1]. Consider the following probabilistic reward function R. For every formula $\phi = \phi(x_1, ..., x_n)$ 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 $r_1, \ldots, r_n$ and set $b = \phi(r_1, \ldots, r_n)$.
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 $2\left(1 - \frac{y}{2^n}\right) - \alpha(y) + 1$.

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 $b = \phi(r_1, ..., r_n)$ 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 $x_1, ..., x_n$ 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.

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

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 $\#SAT \in$ FRMA[1].

Since any problem in #P is reducible to $\#SAT$ under one-to-one reductions, to complete the proof that #P $\subset$ FRMA[1], we need to show that 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 $g \in$ FRMA[1] and let (α, β, γ) be a one-to-one reduction from some function f to g. We want to show that $f \in$ FRMA[1]. Since $g \in$ 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 $R_f : \{0, 1\}^* \times \{0, 1\}^* \to \mathbb{R}$ such that for all x in the domain of f, $R_f(x, y) = R_g(w, z)$ where $w = \alpha(x), z = \beta(y)$. Since R_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

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

$$\mathbb{E}R_g(w, z) = \mathbb{E}R_g(\alpha(x), \beta(y)) = \mathbb{E}R_f(x, y).$$

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 $f \in$ FRMA[1] and that this class is closed under one-to-one reductions.

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

Rational Proofs and Zero Knowledge Proofs.

We remark that our rational proof for $\#P$ is also a *zero knowledge proof*. Informally, this is because the only message that the verifier receives from the prover is $y = f(x)$, the output of the function. The verifier learns no additional information about the function f or about the process that the expert used to compute it.

Rational Proofs and Computationally Sounds Proofs.

Our rational proof for $\#P$ is *computationally sound* [33, 35]. In these proofs, the expert needs to give a short (usually polylogarithmic in length) certificate of the correctness of the computation. For our rational proofs for $\#P$, this certificate is short because its length is zero! Merlin only needs to give the answer $y = f(x)$ to the problem he is asked to solve. There is no extra overhead in communication between prover and verifier.

Rational Proofs and Computationally Bounded provers.

In our rational proof for $\#P$, Merlin does not need to be infinitely powerful. In fact, he is only required to be able to solve problems in $\#P$. Furthermore, there is no computational overhead for Merlin. He does not need to perform any more computations than those required to evaluate the function f.

3.1 PP and Discrete Rational Merlin Arthur games

The classical analogue of FRMA[1] is the complexity class MA of Merlin Arthur games. It should be intuitive that, since #P $\subset$ 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 FRMA[1].

DEFINITION 2. *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.*

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 $MAJSAT \in$ PP (as we will show in section **??**) 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 a discrete analogue of theorem 1.

THEOREM 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 $\text{argmax}_y \mathbb{E}R(\phi, y) = \#\phi$, we have that if $\phi \in MAJSAT$, then $\pi(\phi, \text{argmax}_y \mathbb{E}R(\phi, y)) = 1$, and if $\phi \notin MAJSAT$, then $\rho(\phi, \text{argmax}_Y \mathbb{E}R(\phi, y)) = 1$. 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 $\phi \notin MAJSAT$.

Rational Proofs for the Polynomial Hierarchy.

PP is considered to be more powerful than the polynomial hierarchy because Toda's theorem [41] tells us that $PH \subset P^{PP}$. However, it is not known whether $PH \subset PP$. Nevertheless, we can give one round rational proofs for any problem in the polynomial hierarchy by noting the following two facts

1. Toda's Theorem [41] : Any language in PH can be decided by a polynomial time machine that makes one query to $\#P$.

2. Any language L in $P^{\#P[1]}$ (where the polynomial time machine makes only one query to $\#P$) can be decided with a one-round rational proof, where the query to $\#P$ is outsourced to Merlin. As in the proof above, Arthur will ask Merlin for the number of satisfying assignments to a formula ϕ, and reward him using a scoring rule. After Merlin gives us his answer $\#\phi$, we can apply a polynomial time predicate $\pi(x, \#\phi)$ to decide whether $x \in L$.

3.2 Comparison with classical analogues

As mentioned and proved in the introduction, the fact that PP $\subset$ DRMA[1] implies the following corollary, which shows that – under widely believed complexity assumptions – classical Merlin Arthur protocols, introduced by [5], are strictly less powerful than Rational ones.

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

3.3 The crucial role of randomness

For classical Merlin Arthur games, it is widely believed that randomness *does not help*. Indeed, if PrBPP $= Pr$P then MA = NP [30]. 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 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 [3]:

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.*
- *$R : \mathcal{I} \times \mathcal{S} \to \mathbb{R}$ 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 [3]. 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 $P, V : \{0,1\}^* \to \{0,1\}^*$. 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* $P(\mathcal{P}_i) = \text{argmax}_{a_i} \mathbb{E}_{\mathcal{T}_k}[R(\mathcal{T}_k)|a_i, \mathcal{P}_i]$, *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* $DRMA = \bigcup_{k:k>0} DRMA[k]$.

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 section 5. 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 section 6

5. PROVING DRMA = CH

We prove in this section that DRMA = CH. Before we prove this, we recall the definition of the counting hierarchy, as well as a useful result, due to Toran [42]. We will need some details not in his paper, which is why we reprove this result here.

5.1 The Counting Hierarchy

The counting hierarchy CH was introduced by [43] as a counting analogue to the polynomial hierarchy. First, we recall the definition of this hierarchy. Our definitions follow the ones in [1]

DEFINITION 7 (COUNTING OPERATOR [1][43]). *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 8 (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 [42] showed that $\text{CP}_k = \text{PP}^{\text{CP}_{k-1}}$ 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 1 (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 $L \in \text{CP}_k$. This means that there exists a polynomial p and a language $A \in \text{CP}_{k-1}$ 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 $M^A \in \text{PP}^A$ 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 $M^A \in \text{PP}^A$, 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 2. $CH \subset DRMA$

PROOF. We proceed by induction, showing that $\text{CP}_k \subset DRMA[k]$. 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 PP $\subset$ DRMA[1]. Note that DRMA[1] = 1 − DRMA.

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

Consider a language $L \in \mathrm{CP}_k$. Let $M^{\mathrm{CP}_{k-1}}$ be a nondeterministic 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 $M^{\mathrm{CP}_{k-1}}$ 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 $\mathtt{BSR}(Y,b) = Y(b) - \|Y\|^2 + 1$.[4]

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 $M^{\mathrm{CP}_{k-1}}$. Since Merlin is rewarded using the scoring rule $\mathtt{BSR}(Y,b)$, 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 $M^{\mathrm{CP}_{k-1}}$. Note that it does not matter if Arthur reveals his coin tosses (the random path he chooses in the machine $M^{\mathrm{CP}_{k-1}}$).

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 $R(Y,b) = \mathtt{BSR}(Y,b)$

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 $\Delta = \min_{Y \neq B} \|\mathbb{E}_b[\mathtt{BSR}(B,b) - \mathtt{BSR}(Y,b)]\|$. 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 $x_1, ..., x_k$ be Arthur's queries, and let $b_1, ..., b_k$ be the random samples he draws for each query. Let $y_1, ..., y_k$ be Merlin's answers and let $Y_1, ..., Y_k$ be the corresponding random variables in $\{0,1\}^k$. Merlin's reward is given by

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

where $\gamma = \frac{\Delta}{1+\Delta}$ is chosen so that $\sum_{i=2}^{k} \gamma^{i-1} < \frac{\gamma}{1-\gamma} \leq \Delta$.[5]

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 CH $\subset$ DRMA. We now show the opposite direction. We will use the following definition and facts in the proof.

Definition 9. *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. $R : \mathcal{I} \times \mathcal{S} \to \mathbb{R}$ *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* $y^* = \mathrm{argmax}_y \, \mathbb{E}R(x,y)$.

Lemma 3. *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.

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

[5]This technique is inspired by the discounted market scoring rules of [14], 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.

2. *For any oracle A, $NP^A \subset PP^A$.*

3. *$PP^{\#P}$ 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 $P^{\#P}$.*

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 [3]). 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 NP $\subset$ PP. Furthermore, the machine model in both cases is the same: non-deterministic Turing machines. Thus $\mathrm{NP}^A \subset \mathrm{PP}^A$.

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, $PP^{\#\mathrm{P}} = PP^{\mathrm{PP}}$.

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 $P^{\#\mathrm{P}}$. Let $y_k \cdot 2^k + ... + y_1 \cdot 2 + y_0 = f(x)$ be the output of $f(x)$ written in binary. The output bits $y_k, ..., y_0$ are random variables that depend on the sequence of coin tosses r. By linearity of expectation $\mathbb{E}_r[\sum_{i=0}^k y_i 2^i] = \sum_{i=1}^k \mathbb{E}_r[y_i] \cdot 2^i$. Thus, it suffices to compute the expectation of each bit. Let $M_i(x)$ be a probabilistic polynomial time algorithm that tosses $poly(|x|)$ 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 $2^{poly(|x|)}$. 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 DRMA $\subset$ CH.

Lemma 4. *$DRMA \subset CH$*

Proof. We proceed by first showing that DRMA[1] $\subset$ $P^{\mathrm{PP}^{\#\mathrm{P}}} \subset PP^{\mathrm{PP}^{\mathrm{PP}}} = \mathrm{CP}_3$. Our proof then builds on this fact inductively.

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

1. Given x, let $y^* \in \mathrm{argmax}_y \mathbb{E}_r[R(x,y;r)]$ 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 $y^* = \mathrm{argmax}_y \mathbb{E}_r R(x,y;r)$ where r are the random coin tosses of R. This problem is in $\mathrm{NPO}^{\#\mathrm{P}}$ because for every x, y there exists an algorithm in $P^{\#\mathrm{P}}$ that computes $E(x,y) =_{def} \mathbb{E}_r R(x,y;r)$. Such an algorithm exists as shown in lemma 3. It is a problem in $\mathrm{NPO}^{\#\mathrm{P}}$ because we want to find y^* maximizing $E(x,y)$ and $E(x,y)$ is in $FP^{\#\mathrm{P}}$.

Since $\mathrm{NPO}^{\#\mathrm{P}}$ is Turing reducible to $\mathrm{NP}^{\#\mathrm{P}}$ (again, by lemma 3), there exists some polynomial time machine $M^{NP^{\#\mathrm{P}}}$ such that $M^{NP^{\#\mathrm{P}}}(x)$ finds a y^* maximizing $E(x,y)$. After finding such a y^*, our machine M evaluates $\pi(x,y^*)$. If $\pi(x,y^*) = 1$ then it outputs $x \in L$. Otherwise it outputs $x \notin L$. This shows that $L \in P^{\mathrm{NP}^{\#\mathrm{P}}} \subset P^{PP^{\mathrm{PP}}} \subset PP^{PP^{PP}} = \mathrm{CP}_3$. Thus, DRMA[1] $\subset \mathrm{CP}_3$.

We now show that DRMA[k] $\subset CP_{2k+1}$, for all $k \geq 1$. Let L be a language in DRMA[k] with associated reward and output functions R and π. Let x be an input and let $\mathcal{T}^* = (a_1, b_1, ..., a_k, b_k)$ be a transcript of Merlin and Arthur's interaction on input x. Without loss of generality, we assume that Arthur's messages $b_1, ..., b_k$ consist exclusively of random coins.[6] Given Arthur's coins and the transcript up to round i, Merlin can compute Arthur's corresponding message in round i.

Thus, the transcript is completely determined by Merlin's messages $(a_1, ..., a_k)$. Merlin determines these messages by backwards induction as follows.

- Let $\mathcal{T}_{k-1} = (a_1, b_1, ..., a_{k-1}, b_{k-1})$ be the transcript up to time $k-1$, let a_k be a possible message from Merlin at time k and let $b_k \leftarrow \{0,1\}^{poly(n)}$ be Arthur's random coins at time k. Let $\mathcal{T} = [\mathcal{T}_{k-1}, a_k, b_k]$ be the final transcript when Merlin chooses a_k and Arthur tosses coins b_k. Merlin's value form choosing a_k is $V_k(a_k; \mathcal{T}_{k-1}) = \mathbb{E}_{b_k \leftarrow \{0,1\}^{poly(n)}} R(x, \mathcal{T})$. Merlin will choose a message

$$a_k^* \in \operatorname*{argmax}_{a_k} V_k(a_k; \mathcal{T}_{k-1}).$$

  Denote by $V_k^*(\mathcal{T}_{k-1}) = V_k(a_k^*; \mathcal{T}_{k-1})$ the maximum value of this function.

- Let $\mathcal{T}_{i-1} = (a_1, b_1, ..., a_{i-1}, b_{i-1})$ be the transcript up to time $i-1$ and let $\mathcal{T} = [\mathcal{T}_{i-1}, a_i, b_i, ..., a_k, b_k]$ be a possible final transcript. Note that $a_j(\mathcal{T}_{j-1})$ is a function of the transcript up to time j, and thus, each a_j for $j > i$ is a function of a_i. Merlin's value from choosing a_i is $V_i(a_i; \mathcal{T}_{i-1}) = \mathbb{E}_{b_i \leftarrow \{0,1\}^{poly(n)}} V_{i+1}^*(\mathcal{T}_i)$ where $V_{i+1}^*(\mathcal{T}_i) = \max_{a_{i+1}} \mathbb{E}_{b_{i+1}} \max_{a_{i+2}} \ldots \mathbb{E}_{b_k} R(x, \mathcal{T})$ is a function of the transcript $\mathcal{T}_i$ up to time i. Merlin chooses a_i^* to satisfy

$$a_i^* \in argmax_{a_i} V_i(a_i; \mathcal{T}_{i-1}).$$

  Denote by $V_i^*(\mathcal{T}_{i-1}) = V_i(a_i^*; \mathcal{T}_{i-1})$ the maximum value of this function.

[6] We are relying on the fact that our protocols are public coin protocols.

We want to simulate Merlin using the counting hierarchy as an oracle. We begin by simulating Merlin's choice of his last message a_k. The function $V_k(a_k; \mathcal{T}_{k-1}) = \mathbb{E}_{b_k \leftarrow \{0,1\}^{poly(n)}} R(x, \mathcal{T})$ can be computed in $FP^{\#P}$ according to lemma 3 and the fact that $R(\cdot,\cdot)$ is polynomial time computable. Making a polynomial number of queries to an $NP^{\#P}$ machine, we can perform binary search over the inputs of $V_k(\cdot; \mathcal{T}_{k-1})$ to find a_k^* that maximizes this function. Since $NP^{\#P} \subset PP^{PP}$, we can find a_k^* with access to a CP_2 oracle.

Now we proceed by induction to show that we can find $a_i^*, a_{i+1}^*, ..., a_k^*$ using queries to $CP_{2\cdot(k-i+1)}$. The base case was proved in the preceding paragraph. Assume inductively that, for any transcript prefix $\mathcal{T}_i$, we can find an optimal continuation $a_{i+1}^*, ..., a_k^*$ using queries to $CP_{2\cdot(k-i)}$.

Given input $\mathcal{T}_{i-1}$, we would like to find an optimal continuation $a_i^*, a_{i+1}^*, ..., a_k^*$ using queries to $CP_{2\cdot(k-i+1)}$. It suffices to find a_i^*, because by inductive hypothesis, given $\mathcal{T}_{i-1}, a_i^*$ and random coins b_i, we can compute an optimal continuation $a_{i+1}^*, ..., a_k^*$ using queries to $CP_{2\cdot(k-i)} \subset CP_{2\cdot(k-i+1)}$.

We show that $a_i^*(\mathcal{T}_{i-1})$ can be computed using queries to $NP^{\#P^{CP_{2\cdot(k-i)}}} \subset CP_{2\cdot(k-i+1)}$. First of all, the function $V_i(a_i, \mathcal{T}_{i-1}) = \mathbb{E}_{b_i} V_{i+1}^*(\mathcal{T}_i)$ can be computed using a polynomial number of queries to $\#P^{\mathrm{CP}_{2\cdot(k-i)}}$. Intuitively, the idea is that the $\mathrm{CP}_{2\cdot(k-i)}$ oracle is used to compute $V_{i+1}^*(\mathcal{T}_i)$ for each possible value of the random coin tosses b_i. Each branch of the $\#P$ machine computes $V_{i+1}^*(\mathcal{T}_i)$ for a different setting of b_i, and then adds the results in order to get the expectation. Of course, this computation needs to be done bit-by-bit as in lemma 3.

Since $V_i(a_i, \mathcal{T}_{i-1})$ can be computed using queries to $\#P^{CP_{2\cdot(k-i)}}$, an a_i^* that maximizes $V_i(\cdot, \mathcal{T}_{i-1})$ can be computed using queries to $NP^{\#P^{CP_{2\cdot(k-i)}}}$, which is what we wanted to show. We remark that the computed a_i^* depends on the prefix $\mathcal{T}_{i-1}$.

Carrying out the induction up to $i = 1$, we obtain that we can compute $a_1^*, a_2^*, ..., a_k^*$ using oracle queries to CP_{2k}. Since we need to make polynomially many such queries, and then pass the simulated transcript to the polynomial time predicate $\pi(\cdot,\cdot)$, we conclude that any language in DRMA[k] can be decided in $P^{CP_{2k}} \subset PP^{CP_{2k}} = CP_{2k+1}$.

Q.E.D.

6. PROVING RIP = 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 5. *RIP = PSPACE*

PROOF. We first show RIP $\subset$ PSPACE. Let $L \in$ 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 $i = 1...k = poly(|x|)$

   (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 it's 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 5. *PSPACE $\subset$ RIP.*

PROOF. Since IP = PSPACE [39], 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 $x \notin L$.[7] Thus, L admits a rational interactive proof with polynomially many rounds.

Q.E.D.

7. NEW SCORING RULES

For our application, the existing Brier scoring rule is good enough because we only consider a world with two states $\{0, 1\}$. We would like however to mention some new results on scoring rules that may be of separate interest.

General scoring rules over more complicated state spaces have disadvantages that make them hard to use for computational applications. In particular,

[7]It 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.

(a) *(The range of) S might be unbounded.*
In this case one of the parties may not have enough money to make the required payment. For instance, in Good's rule $\log \mathcal{D}(\omega)$ may be an arbitrarily low negative number, and thus $Merlin$ may have to make an arbitrarily high payment: hardly a reward for his help!

(b) *Evaluating S may be computationally hard.*
For instance, when the set Ω of states of the world contains too many elements, computing $\sum_{\omega \in \Omega} \mathcal{D}(\omega)^2$, as required by Brier's rule, may be too hard!

Thus, there is a tradeoff between a scoring rule being bounded and its query complexity. We have shown that this tradeoff always holds when the scoring rule queries the probability density function of the distribution $\mathcal{D}$. This is what all known scoring rules do. However, if we allow the scoring rule to query the *cumulative density function* of the distribution, then we can obtain new scoring rules that have bounded range, make few queries to their distribution, and are strictly proper. In particular

(a) All known proper scoring rules which have bounded range require more than $\frac{|\Omega|}{4}$ queries to the probability density function of $\mathcal{D}$.

(b) There exists a proper scoring rule which has bounded range and requires $O(\log |\Omega|)$ queries to the cumulative density function of $\mathcal{D}$.

8. SUMMARY OF CONTRIBUTIONS AND FUTURE DIRECTIONS

Our contributions are a new type of proof system and a characterization of said proof system in terms of the counting hierarchy.

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 [35, 33, 31, 25, 13, 7, 36, 9, 22, 15, 16, 2, 10, 38, 17]. 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 $\#SAT \in$ 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. We thank Lance Fortnow for pointing out an error in our original proof of $RMA \subset CH$.

9. 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] 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.

[4] L. Babai, L. Fortnow, and C. Lund. Non-deterministic exponential time has two-prover interactive protocols. *Computational complexity*, 1(1):3–40, 1991.

[5] 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.

[6] 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.

[7] B. Barak and O. Goldreich. Universal arguments and their applications. *ccc*, page 0194, 2002.

[8] 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.

[9] S. Benabbas, R. Gennaro, and Y. Vahlis. Verifiable delegation of computation over large datasets. 2011.

[10] 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 Theoretial Computer Science, 2012.

[11] R.B. Boppana and J. Hastad Stathis. Does-np have short interactive proofs? *Information Processing Letters*, 25(2):127–132, 1987.

[12] G.W. Brier. Verification of forecasts expressed in terms of probability. *Monthly weather review*, 1950.

[13] 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.

[14] 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.

[15] 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.

[16] G. Cormode, J. Thaler, and K. Yi. Verifying computations with streaming interactive proofs.

[17] Ivan Damgård, Sebastian Faust, and Carmit Hazay. Secure two-party computation with low communication. Cryptology ePrint Archive, Report 2011/508, 2011.

[18] U. Feige and J. Kilian. Making games short. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 506–516. ACM, 1997.

[19] L. Fortnow, J. Rompel, and M. Sipser. On the power of multi-prover interactive protocols. *Theoretical Computer Science*, 134(2):545–557, 1994.

[20] L. Fortnow and R.V. Vohra. The complexity of forecast testing. *Econometrica*, 77(1):93–105, 2009.

[21] Lance Fortnow and Michael Sipser. Are there interactive protocols for co-np languages? *Information Processing Letters*, 28:249–251, 1988.

[22] 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.

[23] T. Gneiting and A.E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007.

[24] 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.

[25] 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.

[26] 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.

[27] 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.

[28] I.J. Good. Rational decisions. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 107–114, 1952.

[29] 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.

[30] 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.

[31] Y. Kalai and R. Raz. Probabilistically checkable arguments. *Advances in Cryptology-CRYPTO 2009*, pages 143–159, 2009.

[32] J. Katz. Bridging game theory and cryptography: Recent results and future directions. *Theory of Cryptography*, pages 251–272, 2008.

[33] 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.

[34] 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.

[35] S. Micali. Computationally sound proofs. *SIAM Journal on Computing*, 30(4):1253–1298, 2001.

[36] C. Papamanthou, R. Tamassia, and N. Triandopoulos. Optimal verification of operations on dynamic sets. In *Proc. International Cryptology Conference (CRYPTO)*, 2011.

[37] A. Sandroni. The reproducible properties of correct forecasts. *International Journal of Game Theory*, 32(1):151–159, 2003.

[38] Aviad Rubinstein Shafi Goldwasser, Huijia Lin. Delegation of computation without rejection problem from designated verifier cs-proofs. Cryptology ePrint Archive, Report 2011/456, 2011.

[39] A. Shamir. Ip= pspace. *Journal of the ACM (JACM)*, 39(4):869–877, 1992.

[40] Y. Shoham and M. Tennenholtz. Non-cooperative computation: Boolean functions with correctness and exclusivity. *Theoretical Computer Science*, 343(1-2):97–113, 2005.

[41] S. Toda. Pp is as hard as the polynomial-time hierarchy. *SIAM Journal on Computing*, 20:865, 1991.

[42] J. Torán. Complexity classes defined by counting quantifiers. *Journal of the ACM (JACM)*, 38(3):752–773, 1991.

[43] K.W. Wagner. The complexity of combinatorial problems with succinct input representation. *Acta Informatica*, 23(3):325–356, 1986.