Electronic Colloquium on Computational Complexity, Report No. 149 (2006)

The Complexity of Forecast Testing

Lance Fortnow* Rakesh V. Vohra†

December 7, 2006

Abstract

Consider a weather forecaster predicting a probability of rain for the next day. We consider tests that given a finite sequence of forecast predictions and outcomes will either pass or fail the forecaster. Sandroni (2003) shows that any test which passes a forecaster who knows the distribution of nature, can be probabilistically passed by a forecaster with no knowledge of future events.

We look at the computational complexity of such forecasters and exhibit a linear-time test and a distribution of nature such that any forecaster without knowledge of the future that can fool the test must be able to solve PSPACE-hard problems by requiring the forecaster to simulate a prover in an arbitrary interactive proof system.

1 Introduction

Suppose one is asked to forecast the probability of rain on successive days. Sans knowledge of the distribution that governs the change in weather, how should one measure the accuracy of the forecast? Amongst the many criteria for judging the effectiveness of a probability forecast, one, called calibration, has been the object of recent interest. Dawid [2] offers the following intuitive definition of calibration:

> Suppose that, in a long (conceptually infinite) sequence of weather forecasts, we look at all those days for which the forecast probability of precipitation was, say, close to some given value ω and (assuming these form an infinite sequence) determine the long run proportion p of such days on which the forecast event (rain) in fact occurred. The plot of p against ω is termed the forecaster's *empirical calibration curve*. If the curve is the diagonal $p = \omega$, the forecaster may be termed (empirically) *well calibrated*.

Foster and Vohra [4] exhibit a randomized forecasting algorithm is derived that with high probability will be calibrated on all sequences of wet-dry days. Thus, a forecaster with no meteorological knowledge would be indistinguishable from a one who knew the distribution that governs the change

*Department of Computer Science, University of Chicago, Chicago, IL 60637.

†Department of Managerial Economics and Decision Sciences, Kellogg Graduate School of Management, Northwestern University, Evanston IL 60208. Research supported in part by NSF grant ITR IIS-0121678.

ISSN 1433-8092

in weather.[1] This has inspired the search for tests of probability forecasts that can distinguish between a forecaster who knows the underlying distribution of the process being forecast from one who simply 'games' the test.

A test takes as input a forecasting algorithm, a sequence of outcomes and after some period accepts the forecast (PASS) or rejects it (FAIL). Sandroni [9] proposes two properties that such a test should have. The first is that the test should declare PASS/FAIL after a finite number of periods. This seems unavoidable for a practical test. Second, suppose the forecast is indeed correct i.e., accurately gives the probability of nature in each round. Then, the test should declare PASS with high probability. We call this second condition "passing the truth." Call a test that satisfies these two conditions a *good* test. A test based on calibration would be a good test. A forecaster with no knowledge of the underlying distribution that can pass a good test with high probability on all sequences of data is said to have *ignorantly* passed the test. For every good test, Sandroni [9] shows that there exists a randomized forecasting algorithm that will ignorantly pass the test. Therefore, no good test can distinguish between a forecaster who knows the underlying distribution of the process being forecast from one who simply 'games' the test.

Dekel and Feinberg [3] as well as Olszewski and Sandroni [7] get around the impossibility of result of Sandroni [9] by relaxing the first property of a good test. For example, allowing the test to declare PASS/FAIL at 'infinity', allowing the test to declare FAIL in a finite number of periods but PASS 'at infinity' or relaxing the condition that the test always passes the truth. These tests are computationally infeasible in a variety of ways. In some cases, the number of steps while finite are not bounded. In other cases they rely on the continuum hypothesis or the axiom of choice.

Olszewski and Sandroni [8] have noted that the tests considered by Dekel and Feinberg [3] and Olszewski and Sandroni [7] rely on counterfactual information. Specifically, the test can use the predictions the forecast would have made along sequences that did not materialize. This is because the test has access to the forecasting algorithm itself. As noted in by Olszewski and Sandroni [8] this is at variance with practice. For this reason they consider tests that are not permitted to make use of counterfactual predictions on the part of the forecaster but relax the condition that the test must decide in finite time. Formally, two different forecasting algorithms that produce the same forecast on a realization must be treated in the same way. If such tests pass the truth with high probability they show that for each such test, there is a forecasting algorithm that can ignorantly pass the test.

The present paper examines the consequences of imposing computational limits on both forecaster and the test. Our first result says, roughly, that no forecasting algorithm can ignorantly pass a good test that is more complex than itself. One can think of the forecaster as an algorithm whose running time grows with n, the length of the sequence observed so far. The complexity of the forecaster is measured by the manner in which its running time grows with n. We suppose the complexity of the forecaster to be $t(n)$, where $t(\cdot)$ is a time constructible function n. This means the value of $t(n)$ can be constructed from n by a Turing machine in time $O(t(n))$. Most good test with complexity $p(n)t(n)$ (here $p(n)$ is a polynomial in n) that for sufficiently large n cannot be ignorantly passed by a forecasting algorithm of complexity $t(n)$. Furthermore, the test does not rely on counterfactual predictions. The result, while unsurprising, highlights that the impossibility

[1] Lehrer [5] and Sandroni [9] give generalizations of this result.

result of [9] arises from the absence of any computational constraints on the forecaster.

Our second result exhibits a good test of complexity linear in n with the following property: a forecasting algorithm that could ignorantly pass this test would provide a probabilistic polynomial time algorithm for factoring composite numbers. The existence of a probabilistic polynomial time algorithm for factoring composite numbers is considered unlikely. Indeed, many commercial available cryptographic schemes are based on just this premise. This result suggests that the 'ignorant' forecaster of [9] must have access to computational resources that beggar belief.

Our third result strengthens the second. We exhibit a good test of complexity linear in n with the following property: a forecasting algorithm that could ignorantly pass this test can be used as as a good prover in an interactive proof system. In particular it must solve PSPACE-hard problems. In complexity theory the class PSPACE, is the set of decision problems that can be solved by a deterministic or nondeterministic Turing machine using a polynomial amount of memory and unlimited time. They contain the more well known class of NP problems. Hence, the 'ignorant' forecasts of [9] must be exceedingly powerful.

The idea behind the proofs of the last two results is this. We can interpret the observed sequence of 0-1's as encoding potential answers to some computational problem. For example, the sequence may encode a number followed by a list of its possible factors. Only some of these sequences correspond to correct answers to our computational problem. The test fails any forecaster that fails to assign high probability to these correct sequences when they are realized. Consider now the distribution that puts most of its weight on the sequences that correspond to correct answers. If the forecaster can ignorantly pass the test, it must be able to identify the sequences that correspond to correct answers to our computational question.

In Section 2 we give formal definitions of forecasters and tests. Section 3 shows that the forecaster needs to be nearly as powerful as the test. Section 4 gives a simple test where a successful ignorant forecasters must be able to factor. In Section 5 we sketch a more complicated test that checks if the forecaster successfully acts like a prover in an interactive proof system. This section will also provide a brief description of an interactive proof system and its implications. We draw conclusions and discuss future directions in Section 6.

2 Definitions

Let N be the set of Natural numbers. Let $S = \{0, 1\}$ be the state space.[2] An element of S is called an *outcome*. Let S^n, for $n \in N$, be the n-Cartesian product of S. An n-sequence of outcomes is denoted $s = (s_1, s_2, \ldots, s_n) \in S^n$ where s_i denotes the state realized in period i. Given $s \in S^n$ and $r < n$, let $s^r = (s_1, s_2, \ldots, s_t) \in S^r$ be the prefix of length r of s.

An element of $[0, 1]$ is called a *forecast* of the event '1'. A forecast made at period r refers to outcomes that will be observed in period $r + 1$. Let Δ^* the set of probability distributions over $[0, 1]$. A forecasting algorithm is a function

$$F : \bigcup_{r=0}^{n-1} (S^{r+1} \times [0, 1]^r) \to \Delta^*$$

[2]The results can easily be extended to more than two states.

At the end of each stage $r < n$, a r-history $(s^r, f_0, f_1, \ldots, f_{r-1}) \in S^r \times [0,1]^r$ is observed. Here $f_j \in [0,1]$ is the forecast made by F in period j. Let $f^r = (f_0, \ldots, f_r)$. Based on this r-history, the forecaster must decide which forecast $f_r \in [0,1]$ to make in period r. The forecaster is allowed to randomize. So, $f_r \in [0,1]$ can be selected (possibly) at random, using a probability distribution in Δ^*.

An n-outcome sequence $s \in S^n$ and a forecasting algorithm F determine a probability measure $\bar{F}^s$ on $[0,1]^n$, where conditional on (s^r, f^{r-1}), the probabilities of forecasts next period are given by $F(s^r, f^{r-1})$. The vector of realized forecasts associated with F on a sequence s will be denoted $f(s)$.

Denote the unknown data generating process by $P \in [0,1]^n$. Given $P \in [0,1]^n$ and $s^r \in [0,1]^r$ let $P_{s^r} \in [0,1]$ be the probability that $s_{r+1} = 1$ conditional on s^r. Given P let $F^P(s) \in [0,1]^n$ be the forecast sequence such that $f_r^P(s) = P_{s^r}$.

A *test* is a function $T : S^n \times [0,1]^n \to \{0,1\}$. After a history of n forecasts and outcomes are observed, a test must either accept (PASS) or reject (FAIL) the forecast. When the test returns a 0 the test is said to fail the forecast based on the outcome sequence. When the test returns a 1 the test is said to PASS the forecast based on the outcome sequence.

A test is said to pass the truth with probability $1 - \epsilon$ if

$$\Pr_P(\{T(s, F^P(s)) = 1\}) \geq 1 - \epsilon$$

for all $P \in [0,1]^n$.

A test T can be *ignorantly* passed by a forecasting algorithm F with probability $1 - \epsilon$ if for every $s \in S^n$,

$$\Pr_{\bar{F}^s}(\{T(s, f(s)) = 1\}) \geq 1 - \epsilon.$$

Therefore, F can ignorantly pass T if on any sequence of outcomes, the realized forecast sequence will not be failed with probability at least $1 - \epsilon$ (under the distribution induced by the forecasting scheme). A test T is said to fail the forecasting algorithm F on the distribution Q if

$$\Pr_Q(\{s : \Pr_{\bar{F}^s}(\{T(s, f(s)) = 1\}) \geq 1 - \epsilon\}) \leq \epsilon.$$

Theorem 1 (Sandroni's Theorem) *Suppose test T passes the truth with probability $1-\epsilon$. Then, there is a a forecasting algorithm F that can ignorantly pass T with probability $1 - \epsilon$.*

3 Tests More Complex than the Forecast

We show that no forecasting algorithm can ignorantly pass a test that is more complex itself. For notational simplicity our exposition will be limited to deterministic forecasts. The extension to randomized forecasts is straightforward and we outline it at the end of the section.

Let t be a time constructable function and $p(\cdot)$ a polynomial.

Lemma 1 *For every deterministic forecaster F using time $t(n)$ there is a distribution P and test, T, using time $p(n)t(n)$ that*

1. *passes the truth with probability at least $1 - \epsilon$, and*

2. fails F on P.

Proof
Choose n sufficiently large and a sequence $s^* = (s_1^*, s_2^*, s_3^*, \ldots, s_n^*)$ of outcomes such that the probability F assigns to seeing s_j^* given $(s_1^*, \ldots, s_{j-1}^*)$ and f^{j-2} is at most $1/2$ for all $j \leq n$.

To describe the test, let G be any forecasting algorithm and $(s, g(s))$ be a n-history.

1. If $s \neq s^*$ then $T(s, g(s)) = 1$.
2. If $s = s^*$ and the probability that G assigns to s^* is at least ϵ then $T(s, g(s)) = 1$.
3. If $s = s^*$ and the probability that G assigns to s^* is at most ϵ then $T(s, g(s)) = 0$.

The test always passes the truth since the probability of $s = s^*$ under the true distribution will be less than $(1/2)^n \leq \epsilon$ if the test fails. Therefore the probability that $s \neq s^*$ is at least $1 - (1/2)^n \geq 1 - \epsilon$.

Now we exhibit a distribution P such that $\Pr_P(\{s : T(s, f) = 1\}) = 0$. Let P be the distribution that puts measure 1 on s^*. In this case $s = s^*$ and by part (3) of the test, F fails the test. ■

Lemma 2 *Suppose m deterministic forecasting algorithms, $F_1, F_2, \ldots, F_m$ that each use time $t(n)$. Then there is a distribution P and test T that uses $mp(n)t(n)$ time that*

1. *passes the truth with probability at least $1 - \epsilon$*
2. *and fails all $F_1, F_2, \ldots, F_m$ on P.*

Proof
Consider the sequence of integers $1, 1, 2, 1, 2, 3, 1, 2, 3, 4, 1, 2, 3, 4, 5, \ldots$. Let $h(j)$ be the jth element of the sequence. Choose n sufficiently large and a sequence $s^* = (s_1^*, s_2^*, s_3^*, \ldots, s_n^*)$ of outcomes such that the probability $F_{h(j)}$ assigns to seeing s_j^* given $(s_1^*, \ldots, s_{j-1}^*)$ and $f_{h(j)}^{j-2}$ is $f_{h(j)}(s_j^* | s_1^*, \ldots, s_{j-1}^*, f_{h(j)}^{j-2}) \leq 1/2$ for all $j \leq n$. If $h(j) > m$ let $s_j^* = 0$.

To describe the test, let G be any forecasting algorithm and $(s, g(s))$ be a n-history.

1. If $s \neq s^*$ then $T(s, g(s)) = 1$.
2. If $s = s^*$ and the probability that G assigns to s^* is at least ϵ then $T(s, g(s)) = 1$.
3. If $s = s^*$ and the probability that G assigns to s^* is at most ϵ then $T(s, g(s)) = 0$.

The test always passes the truth since the probability of $\hat{s} = s^*$ under the true distribution will be at most $(1/2)^n < \epsilon$. Therefore probability $\hat{s} \neq s^*$ is at least $1 - \epsilon$.

Let P be the distribution that puts measure 1 on s^*. Choose any F_k. Consider the subsequence $\{s_j^*\}_{h(j)=k}$ of s^*. The probability that F_k assigns to this subsequence is at most $(1/2)^n < \epsilon$. Hence, the failure condition in part (3) of the test holds. ■

Theorem 2 *For any $t(n)$ there is a test T of complexity $p(n)t(n)$ with the following properties.*

1. *For all n the test passes the truth with high probability.*

2. *There is a distribution P on infinite 0-1 sequences such that all forecasts of complexity $t(n)$ fail the test for large enough n.*

3. *P is independent of the forecast and n.*

Proof
Let $F_1, F_2, \ldots$ be an enumeration of all $t(n)$-computable forecasts. Note that the sequence s^* defined in Lemma 2 does not depend on n. Let P be the distribution that puts all its weight on the sequence s^*. For each n the test we require coincides with the test as defined in Lemma 2. As argued in the proof of Lemma 2 this test will pass the truth with high probability for every n. For any $t(n)$ computable test F_k, by Lemma 2, for sufficiently large n, the test will fail F_k on P. ■

To extend the results to randomized forecasts it suffices to show how the proof of Lemma 1 can be modified. Recall that the sequence s^* was chosen so that the forecasted probability assigned by the forecast to s^*_j is less than $1/2$. For a randomized forecast, we choose s^*_j so as to maximize that the probability of choosing a forecast of less than $1/2$ of observing event s^*_j .

4 Forecasts that Factor

In this section we describe a test that always passes the truth and any forecaster that could ignorantly pass this test must be able to factor any number. We prove our results for deterministic forecasters but one can extend, in a manner similar to Section 3, the results to randomized forecasters.

Given an integer k and a 0-1 sequence s, we can interpret the the sequence as encoding an arbitrary tuple of numbers. The first number in the tuple we shall call the prefix of the sequence. The remaining numbers (the suffix) will be interpreted as a potential factorization of the prefix. A sequence that encodes the tuple $(m, \pi_1, e_1, \pi_2, e_2, \ldots, \pi_k, e_k)$ is said to encode the unique factorization of m if

1. $\pi_1, \pi_2, \ldots, \pi_k$ are primes,

2. $\pi_1 > \pi_2 > \ldots > \pi_k > 1$, and

3. $m = \pi_1^{e_1}\pi_2^{e_2}\cdots\pi_k^{e_k}$.

To describe the test, let G be any forecasting algorithm and $(s, g(s))$ be a n-history.

1. Let S^* be the set of all sequences that encode the factorization of a number.

2. If $s \notin S^*$ then $T(s, g(s)) = 1$.

3. If $s \in S^*$, then the prefix of s corresponds to some number m. Determine the probability, p, that G assigns to s conditional on the prefix being m

4. If $p \geq \epsilon$ then $T(s, g(s)) = 1$ otherwise $T(s, g(s)) = 0$.

Call this the *factoring* test. It can be implemented in polynomial-time since we have efficient algorithms for multiplication and primality testing [1]. We can make a linear-time test by appropriately padding the sequence.

Theorem 3 *The factoring test passes the truth with high probability and any forecast algorithm that can ignorantly pass the test can be used to factor composite numbers.*

Proof
Consider any distribution of nature P and the forecaster F^P. Let p_m be the probability that P assigns a sequence whose prefix encodes m. Let q_m be the probability that a sequence encodes the factorization of m conditional on its prefix encoding m. The factoring test will fail F^P if for some m a sequence $s \in S^*$ with prefix m is realized and $q_m < \epsilon$. The probability of this happening is

$$\sum_{m:q_m<\epsilon} p_m q_m < \sum_m p_m \epsilon = \epsilon.$$

Hence, the factoring test will pass the truth with high probability.

Now, fix an m and consider the distribution that puts its full weight on the sequence encoding the unique factorization of m. If the forecaster passes the test for this distribution, the conditional probability that suffix of the sequence reveals the prime factors of m is at least ϵ. Use the forecasted probabilities of the suffix as a distribution to generate a sequence. With probability at least ϵ we generate the factors of m. If we repeat the process $O(1/\epsilon)$ times we determine the factors with high probability. ■

5 PSPACE Hardness

In this section we describe a linear-time test that can be ignorantly passed a forecaster only if they can solve PSPACE-hard problems. Once again we assume deterministic forecasters though the results extend to probabilistic forecasters as well.

In complexity theory the class PSPACE, is the set of decision problems that can be solved by a deterministic or nondeterministic Turing machine using a polynomial amount of memory and unlimited time. They contain the more well known class of NP problems. A logical characterization of PSPACE is that it is the set of problems expressible in second-order logic with the addition of a transitive closure operator. A major result of complexity theory is that PSPACE can be characterized as all the languages recognizable by a particular interactive proof system. The hardest problems within the class PSPACE are called PSPACE-complete.

An interactive proof system is an abstraction that models computation as an exchange of messages between a prover (P) and verifier (V). P would like to convince V that a given x belongs to a given set L.[3] P has unlimited computational resources while V's computational resources are bounded. Messages are passed, in rounds, between P and V until V reaches a conclusion about the correctness of the statement $x \in L$. The parties take turns to send a message to the other party. A strategy for P (or V) is a function that specifies in each round what message is to be sent as

[3]More formally, that a given string x belongs to a certain language L.

function of the history of messages exchanged in prior rounds. A pair of strategies, s_P for P and s_V for V, is called a protocol.

Membership in L admits an interactive proof system if there is a protocol (s_P, s_V) with the following two properties:

1. Completeness: If $x \in L$, and both P and V follow the protocol (s_P, s_V), then V will be convinced that $x \in L$.

2. Soundness: Suppose $x \notin L$. If V follows s_V but P deviates from s_P, V can be convinced that $x \in L$ with some small probability.

Shamir [11], building on work of Lund, Fortnow, Karloff and Nisan [6], exhibits an interactive proof system for any PSPACE language L. The proof system has the following properties for an input x of length n.

- Let F be a field GF(q) for a prime q exponential in n.
- The protocol alternates between the Prover (P) giving a polynomial of degree m over F and the Verifier (V) choosing a random element of F. Both m and the number of rounds r of the protocol are bounded by a polynomial in n.
- V takes all of the messages and decides whether to accept or reject with a deterministic polynomial-time algorithm.
- In each round for V there are at most m *bad* choices from the possible q elements of F that depend only on the previous messages. The other choices we call *good*. It can be computationally difficult to decide whether a choice is good or bad but since $m \ll q$ the verifier will pick good choices with very high probability.
- If $x \notin L$ then for any strategy of P, V will reject if all of its choices are good.
- If $x \in L$ there is a strategy for P that will cause V to accept always. Moreover if all of V's choices are good, the messages of P that cause V to accept are unique.

Fix a PSPACE-complete language L. Consider a sequence of outcomes interpreted as the tuple $(x, \rho_1, v_1, \ldots, v_r, \rho_r)$ where the ρ_i's are P's messages and the v_i's are V's message in an interactive proof system for showing $x \in L$. We call the tuple *accepting* if the verifier would accept if the protocol played out with these messages.

Let p_i be the probabilities that the realized forecaster predicts for each ρ_i and b_i be the probabilities for each v_i.

Given a sequence interpreted as such a tuple and the realized forecasts our test will declare "FAIL" if all of the following occur:

1. $(x, \rho_1, v_1, \ldots, v_r, \rho_r)$ is an accepting tuple,
2. Each b_i is at most $\frac{1}{\sqrt{q}}$, and
3. The product of the ρ_i's is at most $\frac{1}{n}$.

Otherwise the test declares "PASS". The test runs in polynomial time and we can pad the sequence so that the test can run in linear time. Call this the PSPACE test.

Theorem 4 *The PSPACE test passes the truth with high probability. A forecaster that can ignorantly pass the PSPACE test can compute PSPACE problems.*

Proof
Consider any distribution of nature P and the associated forecaster F^P. Let p_i's and b_i's be as described above for F^P. Let d_x be the probability that P puts on choosing a sequence starting with x. By the properties of the proof system we have the probability that the test FAILS is bounded by

$$\sum_x p_x(u_x + w_x)$$

where u_x is the probability that a sequence beginning with x fails the test when all the elements are good and w_x is the probability that a sequence beginning with x fails the test when some element is bad.

The probability w_x is bounded by the probability that some element is bad which is bounded by

$$\sum_i mb_i \leq \frac{mr}{\sqrt{q}} = o(1).$$

If the v_i's are good then there is at most one choice of the ρ_i's that lead to an accepting tuple. So

$$u_x \leq \prod_i p_i \leq \frac{1}{n} = o(1).$$

So the probability that the test fails is

$$\sum_x p_x(u_x + w_x) = o(1) \sum_x p_x = o(1).$$

For any x in L we define the following distribution. Fix a strategy for the prover in the interactive proof system for L that guarantees acceptance for all verifier's messages. For each ρ_i play that strategy, for each v_i pick an element from F uniformly. Output $(x, \rho_1, v_1, \ldots, v_r, \rho_r)$.

Suppose we have a forecaster that passes the test on this distribution with high probability. The first condition of the test is always true. Fix an i. There are q possible v_i of which the forecaster can only put weight greater than $\frac{1}{\sqrt{q}}$ on $\sqrt{q}$ of those possibilities. So with probability at least $1 - r\sqrt{q}/q = 1 - o(1/n)$, all of the b_i's will be less than $\frac{1}{\sqrt{q}}$ and the second condition of the test will be satisfied. By a similar argument, with high probability all of the v_i's are good.

Since the forecaster passes the test with high probability then with high probability the product of the p_i's must be at least $1/n$. So if we use the forecaster as a distribution to generate the prover's responses the verifier will accept with probability close to $1/n$. If we repeat the process $O(n)$ times at least once the verifier will accept with high probability. Whereas if x were not in L repeating the protocol a polynomial number of times will rarely cause the verifier to accept no matter what the prover's strategy. Thus the forecaster gives us a probabilistic algorithm for determining whether x is in L. ■

6 Conclusion

The tests described in sections 4 and 5 suggest that the forecasting algorithms implied by Sandroni's impossibility result must be extremely complex. Our proofs do require that nature can have complicated distributions. However, in Section 4 we can consider the distribution that randomly chooses two large primes p and q and outputs the sequence corresponding to the tuple (pq, p, q) with the same test as before. One can show that any forecaster that passes the test under this distribution can factor products of large primes on average, which would break many current cryptosystems. This would suggest that Sandroni's impossibility result has little relevance to the testing of practical forecasting algorithms.

More generally our results exhibit a difference in the conclusions one can draw in an economic model depending on whether we require the agents involved to have reasonable computational restrictions. We believe many other economic scenarios can also benefit from consideration of computationally bounded agents.

References

[1] M. Agrawal, N. Kayal, and N. Saxena. "PRIMES is in P," *Annals of Mathematics*, 160(2):781–793, 2004.

[2] Dawid, A.P. (1982) "The Well Calibrated Bayesian," *Journal of the American Statistical Association* **77**, 379, 605–613.

[3] Dekel, E. and Y. Feinberg (2006) "Non-Bayesian testing of a Stochastic Prediction," *Review of Economic Studies*, forthcoming.

[4] Foster, D.P. and R.V. Vohra (1998) "Asymptotic Calibration," *Biometrika*, 85-2, 379-390.

[5] Lehrer, E. (2001) "Any Inspection Rule is Manipulable," *Econometrica,* **69**-5, 1333-1347.

[6] C. Lund, L. Fortnow, H. Karloff, and N. Nisan. "Algebraic methods for interactive proof systems," *Journal of the ACM*, 39(4):859–868, 1992.

[7] Olszewski, W. and A. Sandroni (2006) "Strategic Manipulation of Empirical Tests", CMS-EMS discussion paper # 1425.

[8] Olszewski, W. and A. Sandroni (2006) "Counter Factual Predictions", manuscript, Northwestern University.

[9] Sandroni, A. (2003) "The Reproducible Properties of Correct Forecasts," *International Journal of Game Theory*, 32-1, 151-159.

[10] Sandroni, A., R. Smorodinsky and R. Vohra (2003)"Calibration with Many Checking Rules," *Mathematics of Operations Research* 28-1, 141-153.

[11] A. Shamir. "IP = PSPACE," *Journal of the ACM*, 39(4):869–877, 1992.

[12] L. Valiant and V. Vazirani. "NP is as easy as detecting unique solutions," *Theoretical Computer Science* 47(1), 85-93, 1986.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de/