Lower Bounds for Distributed Coin-Flipping and Randomized Consensus*

James Aspnes †

February 2, 1998

Abstract

We examine a class of *collective coin-flipping games* that arises from randomized distributed algorithms with halting failures. In these games, a sequence of *local coin flips* is generated, which must be combined to form a single *global coin flip*. An adversary monitors the game and may attempt to bias its outcome by hiding the result of up to t local coin flips. We show that to guarantee at most constant bias, $\Omega(t^2)$ local coins are needed, even if (a) the local coins can have arbitrary distributions and ranges, (b) the adversary is required to decide immediately whether to hide or reveal each local coin, and (c) the game can detect which local coins have been hidden. If the adversary is permitted to control the outcome of the coin except for cases whose probability is polynomial in t, $\Omega(t^2/\log^2 t)$ local coins are needed. Combining this fact with an extended version of the well-known Fischer-Lynch-Paterson impossibility proof of deterministic consensus, we show that given an adaptive adversary, any t-resilient asynchronous consensus protocol requires $\Omega(t^2/\log^2 t)$ local coin flips in any model that can be simulated deterministically using atomic registers. This gives the first non-trivial lower bound on the total work required by wait-free consensus and is tight to within logarithmic factors.

1 Introduction

Our results divide naturally into two parts: a lower bound for asynchronous randomized consensus in a wide variety of models, and a still more general lower bound for a large class of collective coin-flipping games that forms the basis of the consensus lower bound but is interesting in its own right.

Consensus is a fundamental problem in distributed computing in which a group of processes must agree on a bit despite the interference of an adversary. (An additional condition forbids trivial solutions that always produce the same

*A preliminary version of this paper appeared in STOC '97 [Asp97].

†Yale University, Department of Computer Science, 51 Prospect Street/P.O. Box 208285, New Haven, CT 06520-8285. Supported by NSF grants CCR-9410228 and CCR-9415410. E-mail: aspnes@cs.yale.edu.

answer). In an asynchronous setting, it has long been known that if an adversary can halt a single process, then no deterministic consensus algorithm is possible without the use of powerful synchronization primitives [CIL87, DDS87, FLP85, Her91, LAA87].

In contrast, randomized algorithms can solve consensus in a shared-memory system for n processes even if the adversary can halt up to $n-1$ processes. Such algorithms are called *wait-free* [Her91] because any process can finish the algorithm without waiting for slower (or possibly dead) processes. These algorithms work even under the assumption that failures and the timing of all events in the system are under the control of an *adaptive adversary*— one that can observe and react to all aspects of the system's execution (including the internal states of the processes).

The first known algorithm that solves shared-memory consensus against an adaptive adversary is the exponential-time algorithm of Abrahamson [Abr88]; since its appearance, numerous polynomial-time algorithms have appeared [AH90, ADS89, SSW91, Asp93, DHPW92, BR90, BR91, AW96]. Most of these algorithms are built around *shared coin protocols* in which the processes individually generate many random ± 1 *local coin flips*, which are combined by majority voting. The adversary may bias the outcome of the voting by selectively killing processes that have chosen to vote the "wrong" way before they can reveal their most recent votes to the other processes. To prevent the adversary from getting more than a constant bias, it is necessary to collect enough votes that the hidden votes shift the outcome by no more than a constant number of standard deviations. With up to $n-1$ failures (as in the wait-free case), this requires a total of $\Omega(n^2)$ local coin-flips, and at least $\Omega(n^2)$ work in order to communicate these coin-flips.[1]

Improvements in other aspects of consensus algorithms have steadily brought their costs down, from the $O(n^4)$ total work of [AH90] to the $O(n^2 \log n)$ total work of [BR91]. But while these algorithms have steadily approached the $\Omega(n^2)$ barrier, none have broken it. However, no proof was known that consensus could not be solved in less than $\Omega(n^2)$ time; the barrier was solely a result of the apparent absence of alternatives to using shared coins based on majority voting. Indeed, it was asked in [Asp93] if every consensus protocol contained an embedded shared coin protocol; and (specializing a more general and still open question of Ben-Or and Linial [BOL89]) if no shared coin protocol in this model could beat the $\Omega(n^2)$ cost of majority voting.

[1] Some of the algorithms deviate slightly from the simple majority-voting approach described here. In the algorithm of Aspnes [Asp93], some votes are generated deterministically. In the algorithm of Saks, Shavit, and Woll [SSW91], several coin-flipping protocols optimized for different execution patterns are run in parallel. In the algorithm of Aspnes and Waarts [AW96], processes that have already cast many votes generate votes with increasing weights in order to finish the protocol quickly. However, none of these protocols costs less than simple majority voting in terms of the expected total number of local coin flips performed in the worst case.

1.1 Our Results

We show that for a shared coin protocol to guarantee at most constant bias despite up to t failures, $\Omega(t^2)$ local coins are needed, even if (a) the local coins can have arbitrary distributions and ranges, (b) the adversary is required to decide immediately whether to hide or reveal each local coin, and (c) the protocol can detect which local coins have been hidden. If the protocol has polynomial bias, meaning that the adversary is permitted to control the outcome of the protocol except for cases whose probability is polynomial in t, $\Omega(t^2/\log^2 t)$ local coins are needed. An extended version of the well-known Fischer-Lynch-Paterson impossibility proof of deterministic consensus is then used to show that given an adaptive adversary, any t-resilient asynchronous consensus protocol either executes a shared coin protocol with polynomial bias or carries out an expected $\Omega(t^2)$ local flips avoiding it. This implies that t-resilient asynchronous consensus requires an expected $\Omega(t^2/\log^2 t)$ local coin flips. Since protocols based on majority voting require only $O(t^2)$ local coin flips, this lower bound is very close to being tight.

Since we are counting coin-flips rather than operations, the lower bound is not affected by deterministic simulations. So, for example, it continues to hold in message-passing models with up to t process failures (since a message channel can be simulated by an unboundedly large register), or in a shared-memory model with counters or cheap atomic snapshots. Furthermore, since our lower bound assumes that local coin flips can have arbitrary ranges and distributions, we may assume without loss of generality that any two successive coin-flips by the same process are separated by at least one deterministic operation in any of these models— so the lower bound on local coin-flips in fact implies a lower bound on total work.

The lower bound on coin-flipping games is still more general, and holds in any model in which the adversary may intercept up to t local coin-flips before they are revealed, no matter what (deterministic) synchronization primitives or shared objects are available. Furthermore, it is tight in the sense that it shows that no *constant-bias* shared coin can use less than $\Omega(t^2)$ local coins, a bound achieved by majority voting.

1.2 Related Work

Many varieties of collective coin-flipping games have been studied, starting with the work of Ben-Or and Linial [BOL89]. Many such games assume that the locations of faulty coins are fixed in advance; under these assumptions very efficient games exist [AN90, CL93, BOL89, Sak89]. Another assumption that greatly limits the power of the adversary is to require that both the locations and values of faulty coins are fixed in advance; this is the *bit extraction problem* [CFG+85, Fri92, Vaz85], in which it is possible to derive completely unbiased random bits.

If none of these limiting assumptions are made, the adversary gains considerably more power. If the adversary can subvert running processes based

on the execution of the protocol so far, the best strategy for minimizing the adversary's influence in many models seems to be to take the majority of fair coin-flips, the idea being that the majority function minimizes the influence of any single local coin.[2] Ben-Or and Linial [BOL89] observed that with a restriction to fair coins, Harper's isoperimetric inequality for the hypercube [Har66] implies that the majority function gives the least power to an off-line adversary that can see all coins before deciding which to change (a *one-round* protocol), and conjectured that a similar result held for *multi-round* protocols in which n processes repeatedly executed rounds in which each flipped a coin and the adversary could control all coin-flips of a process once it was subverted.

This conjecture is still open, as the present work applies to systems in which the adversary can only alter one local coin-flip for each process that it subverts. (One can think of this restriction as assuming halting failures rather than Byzantine failures in the processes.) A previous paper with similar scope was that of Lichtenstein, Linial, and Saks [LLS89], who showed that majority is optimal under the assumption of *fair* local coins in a sequential game similar to the one we consider here. In their model fair coin-flips are generated one at a time and the adversary may replace up to k of them, its decisions depending only on the values of the coin-flips generated so far. The main difference between their results and ours are: (a) they require fair Boolean-valued local coins, where we allow arbitrary distributions and ranges on the local coins; (b) they allow the adversary to replace a coin-flip with a new value of its choosing, where we assume a weaker adversary that can only hide a coin-flip by replacing it with a fixed value $\perp$; and (c) they obtain a tight result that shows that combining the local coins with the majority function (or, in general, any threshold function) minimizes the adversary's influence over the global coin. This result depends strongly on the assumption of fair local coins and the techniques used to prove it do not appear to generalize to arbitrary distributions on the local coins.

In contrast, our results work for arbitrary distributions, but we do not resolve completely the question of whether majority is optimal in our more general model. We do show that for constant bias the number of faults cannot exceed $O(\sqrt{n})$, the number tolerated (modulo constant factors) by majority, but for large biases our lower bound diverges from the upper bound given by majority. We believe that a strengthened version of our lower bound could show that majority is asymptotically optimal; this issue is discussed in Section 4.

The best previously known bound for arbitrary local coins is a bound of $\Omega(1/\sqrt{n})$ on the influence of an adversary that can hide *one* coin, due to Cleve and Impagliazzo [CI93]. They show that in any martingale sequence starting at 0 and ending at ± 1, with at least constant probability there is a jump of at least $\Omega(1/\sqrt{n})$. To translate this into a result about coin-flipping, one constructs a martingale $X_0, X_1, \ldots X_n$ by letting X_i be the conditional expectation of the global coin given the values of the first i coins, and observes that if there is a large jump between X_i and X_{i+1} the adversary can get a large influence over

[2] An excellent survey of results for a wide variety of models involving fair or nearly fair two-valued local coins can be found in [BOLS87].

the outcome of the game by hiding the $(i+1)$-th local coin.

Part of the motivation for our work on coin-flipping games was to show a lower bound on the work used by wait-free shared-memory consensus. A very nice lower bound on the *space* used by wait-free shared-memory consensus is due to Fich, Herlihy, and Shavit [FHS93]. They show that any such consensus protocol must use $\Omega(\sqrt{n})$ distinct registers to guarantee agreement. Unfortunately, their techniques do not appear to generalize to showing lower bounds on work.

2 Coin-Flipping Games

A collective coin-flipping game [BOL89] is an algorithm for combining many *local coins* into a single *global coin*, whose bias should be small even though some of the local coins may be obscured by a malicious adversary. Though the particular coin-flipping games we consider here are motivated by their application to proving lower bounds on distributed algorithms with failures, they abstract away almost all of the details of the original distributed systems and are thus likely to be useful in other contexts.

We assume that the local coins are independent random variables whose ranges and distributions are arbitrary. The values of these variables are revealed one at a time to an adversary who must immediately choose whether to reveal or obscure each value. If the adversary chooses to obscure the value of a particular local coin, the effect is to replace it with a default value $\perp$. Repeating this process yields a sequence of values, some of which are the original values of the random variables and some of which are $\perp$. A function is applied to this sequence to yield an *outcome*, which may be arbitrary but which we will usually require to be ± 1. The adversary's power is limited by an upper bound on how many coins it may obscure.

Note that in this description we assume that the adversary cannot predict future local coins; it can only base the decision to reveal or obscure a particular coin on the coin's value and the values of earlier coins. In addition, the adversary's interventions are visible. The coin-flipping game may observe and react to the fact that the adversary has chosen to obscure particular local coins, even though it has no access to the true values of those coins.

Formally, a coin-flipping game is specified by a tree. The leaves of the tree specify the outcomes of the game. Internal nodes correspond to local coin-flips. Coin-flipping games are defined recursively as follows. Fix a set of possible outcomes. A coin-flipping game G with maximum length zero consists of a single outcome; we will call such a game a *constant game* and abuse notation by writing its outcome simply as G. A coin-flipping game G with maximum length n is either a constant game or consists of

1. A random variable representing the first local coin-flip in G.
2. A function mapping the range of this random variable to the set of coin-flipping games with maximum length less than n (the *subgames* of G). For

each value α in this range, the resulting subgame is denoted G_α.

3. A *default subgame* $G_\perp$ with maximum length less than n, corresponding to the effect of an adversary choice to hide the first local coin-flip in G.

The above definition represents a coin-flipping game as a tree; if we think of G as the root of the tree its children are the subgames G_α for each value of α and the default subgame $G_\perp$. The actual game tree corresponding to playing the game against an adversary is a bit more complicated and involves two plies for each level of G. We may think of the states of this game as pairs (G, k) specifying the current subgame G and the limit k on how many local coins the adversary may hide (i.e., the number of *faults*). To execute the first local coin-flip in G, two steps occur. First, the outcome α of the coin-flip is determined. Second, the adversary chooses between revealing α, leading to the state (G_α, k); or hiding α, leading to the state $(G_\perp, k-1)$.

In order to prevent the adversary from being able to predict the future or the game from being able to deduce information about obscured coins, we demand that all random variables on any path through the game tree be independent.

An adversary strategy specifies for each partial sequence of local coin-flips whether to hide or reveal the last coin. We will write $G \circ A$ for the random variable describing the outcome of G when run under the control of an adversary strategy A. If a game G has real-valued outcomes, then for each number of faults k there exist adversary strategies to maximize or minimize the expected outcome. Define $M_k G$ to be the maximum expected outcome and $m_k G$ to be the minimum expected outcome. These values can be computed recursively as follows:

- If G has length 0, $M_k G = m_k G = G$.
- If G has positive length, then

$$M_k(G) = \mathrm{E}_\alpha \left[\max\left(M_k G_\alpha, M_{k-1} G_\perp\right)\right] \quad (1)$$

$$m_k(G) = \mathrm{E}_\alpha \left[\min\left(m_k G_\alpha, m_{k-1} G_\perp\right)\right]. \quad (2)$$

Most of the time we will assume that the only possible outcomes of a game are ± 1. In this case the quantities M_k and m_k give a measure of how much influence an adversary with the ability to hide k local coin-flips can get over the outcome. It is necessary to consider both at once: as we will see later, it is always possible to find a game with maximum length n whose minimum expected outcome m_k can be any value in the range $[-1, 1]$. We will be interested in the best such game, i.e., the one that attains a particular value of m_k while minimizing M_k (or, symmetrically, the game that maximizes m_k for a particular fixed M_k). In general it will turn out to be quite difficult to find this game exactly (although much can be shown about its structure), and so it will be necessary to settle for a lower bound on $M_k G$ as a function of n, k, and $m_k G$.

2.1 The Structure of Optimal Games

Fix a maximum length n and number of failures k. Let us define the *range* of a game G to be the interval $[m_k G, M_k G]$. Then G *(strictly) dominates* G' just in case the range of G is a (proper) subset of the range of G'; in other words, if G gives the adversary no more control than G' does. A game G is *optimal* if it either dominates all other games G' with $m_k G' = m_k G$ or if it dominates all other games G' with $M_k G' = M_k G$. For $k < n$, this definition will turn out to be equivalent to saying that no game strictly dominates G.

With each k and game G we can associate a point in a two-dimensional space given by the coordinates $m_k G$ and $M_k G$. From this geometric perspective the problem we are interested in is finding for each value of n and k the curve corresponding to the set of optimal games with maximum length n and up to k failures.

For some values of n and k this task is an easy one. If $k = 0$, then the $(n, 0)$ curve is just the diagonal running from $(-1, -1)$ to $(1, 1)$, since $m_0 G = M_0 G$ for all G. If the other extreme holds and $k \geq n$, then for any G either $m_k G = -1$ or $M_k G = 1$, depending on the default outcome of G if all local coins are hidden. It is not difficult to see that if $M_n G = 1$, then $m_n G$ can be any value between -1 and 1. For example, G could set its outcome to be the value of the first local coin, or 1 if that coin-flip is hidden; if the adversary wishes to achieve an outcome lower than 1 it must let the first local coin go through. Similar, if $m_n G = -1$ then $M_n G$ can be any value between -1 and 1. Thus the optimal (n, n) curve consists of the line segment from $(-1, -1)$ to $(-1, 1)$ and the line segment from $(-1, 1)$ to $(1, 1)$.

Equations (1) and (2) have a nice geometrical interpretation that in principle allows one to determine the (n, k) curves of optimal games of maximum length n with k failures. This process is depicted in Figures 1 and 2. Fix a game G. Each subgame G_α corresponds to a point $(m_k G_\alpha, M_k G_\alpha)$, which must lie somewhere on or above the curve of optimal $(n-1, k)$ games. The contribution of G_α to the position of G is given by $(\min(m_k G_\alpha, m_{k-1} G_\perp), \max(M_k G_\alpha, M_{k-1} G_\perp))$, which is a point in the intersection of the region above the $(n-1, k)$ curve and the rectangle of points dominated by $G_\perp$. Since the value of G is the average of these contributions, it must correspond to some point in the convex closure of this intersection. Provided the $(n-1, k)$ curve is concave (which is easily proved by induction on n as shown below), then all points in the convex closure are dominated by some point on its lower right edge: the line segment between the optimal $(n-1, k)$ game G_0 with $M_k G_0 = M_{k-1} G_\perp$ and the optimal $(n-1, k)$ game G_1 with $m_k G_1 = m_{k-1} G_\perp$.

Geometrically, this edge is the hypotenuse of a right triangle inscribed between the $(n-1, k)$ and $(n-1, k-1)$ curves such that its sides are parallel to the axes and its right corner is on the $(n-1, k-1)$ curve. To take into account all possible choices of $G_\perp$, it is necessary to consider all such triangles. By taking the minimum of the hypotenuses of these triangles (as shown in Figure 2), we obtain the (n, k) curve of all optimal games of maximum length n subject to up to k failures. Note that if the $(n-1, k)$ curve is nondecreasing and concave (true

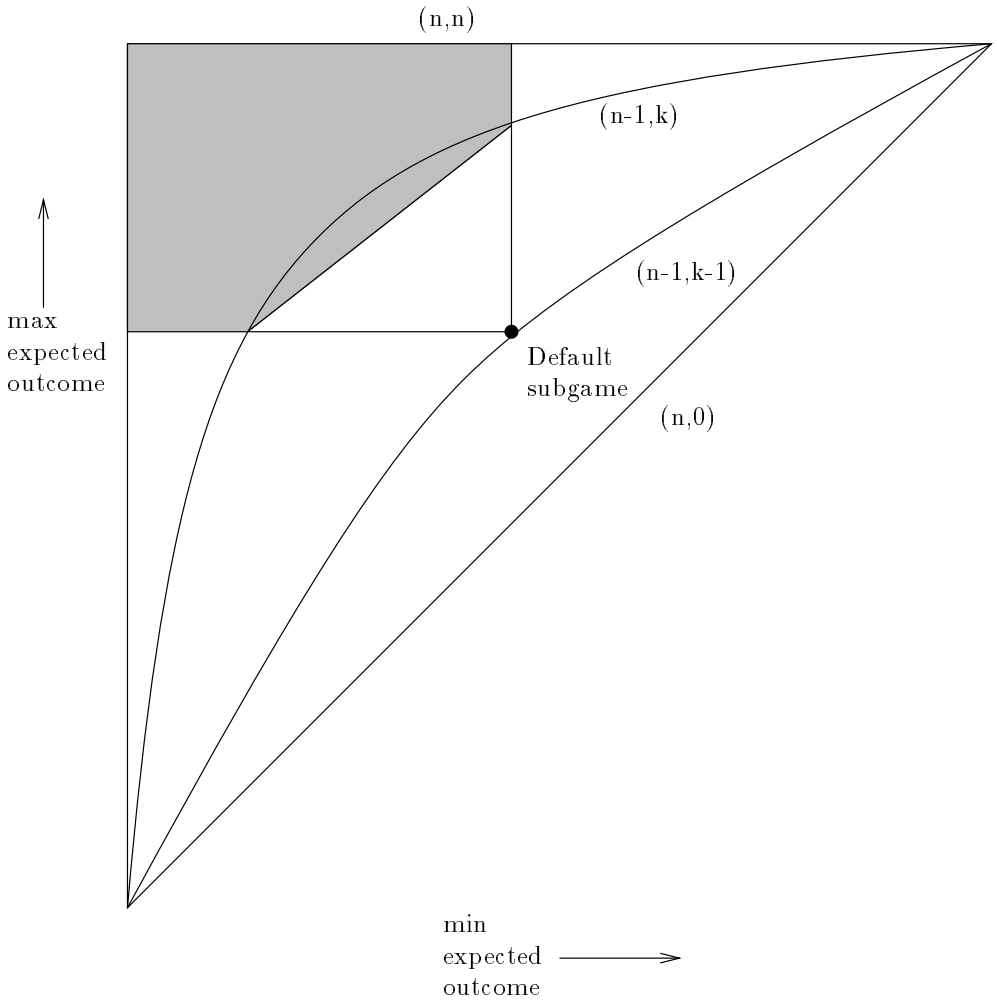

Figure 1: Graphical depiction of constraints on minimum and maximum expected outcomes of a game G given n and k. Each point in the figure corresponds to a pair of minimum and maximum expected outcomes. The diagonal represents the $k = 0$ case where these values are the same. The outer edges of the figure represent the $k = n$ case. The two inner curves represent all optimal games with $n - 1$ voters and either k or $k - 1$ failures. The default subgame $G_\perp$ lies somewhere on or above the $(n - 1, k - 1)$ curve. All other subgames G_α lie on or above the $(n - 1, k)$ curve. If $G_\perp$ is fixed, the value of G lies somewhere in the convex closure of the intersection of the region above the $(n - 1, k)$ curve and the rectangle dominated by $G_\perp$. All points in this convex closure, shown shaded in the picture, are dominated by some point on the hypotenuse of the right triangle inscribed between the $(n - 1, k)$ and $(n - 1, k - 1)$ curves.

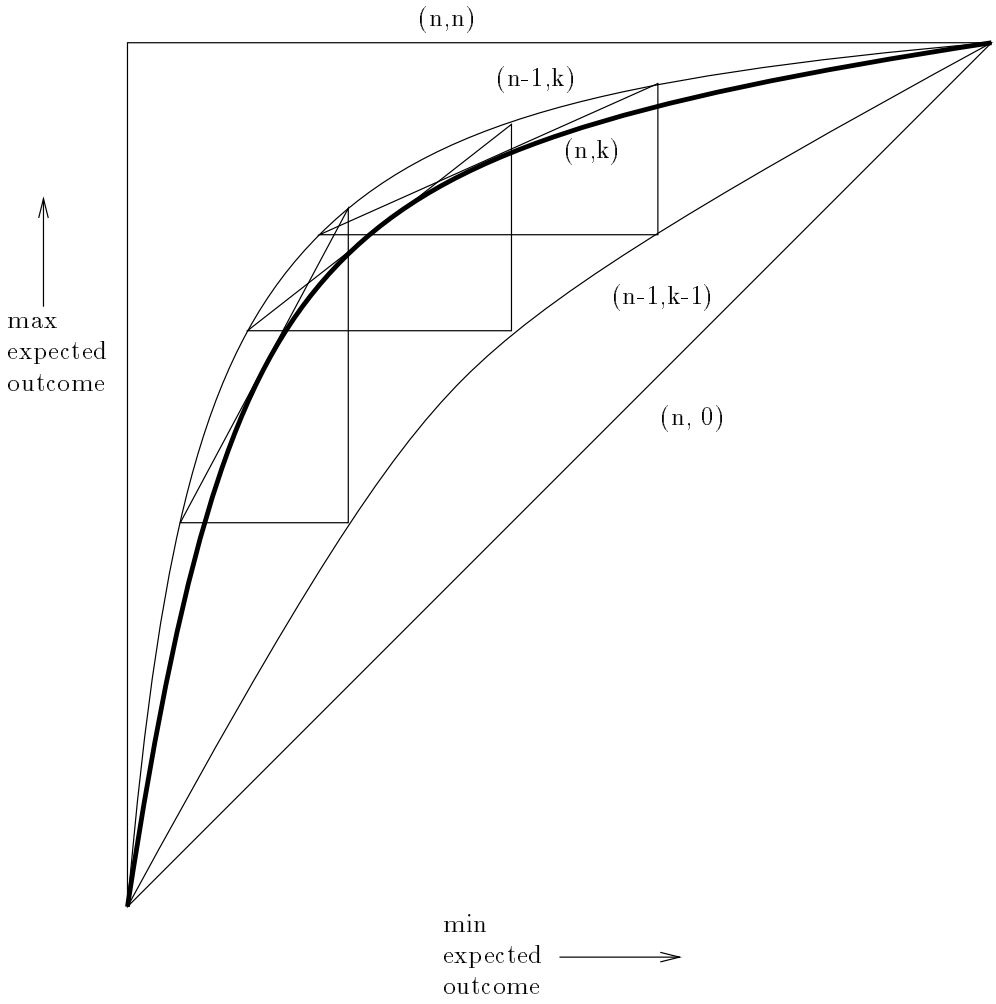

Figure 2: Effect of considering all choices of $G_\perp$. Each point on the $(n-1,k-1)$ curve corresponds to some possible default subgame $G_\perp$. The hypotenuse of the right triangle with corners on this point and the $(n-1,k)$ curve gives a set of games which dominate all other games with this fixed $G_\perp$. The set of optimal games with n voters and k failures is thus the minimum of the hypotenuses of all such right triangles.

for $n-1=k$, true as the induction hypothesis for larger $n-1$), we may extend each hypotenuse to its containing line without affecting the minimum, and so the (n,k) curve as the minimum of concave functions is also nondecreasing and concave.

Let us summarize. From the discussion of the constraints on G given $G_\perp$, we have:

Theorem 1 *For each coin-flipping game G with maximum length n and up to k failures, there is a G' such that G' dominates G, $G'_\perp$ dominates $G_\perp$, G' has exactly two non-default subgames G'_0 and G'_1, $M_k G'_0 = M_{k-1} G'_\perp$, and $m_k G'_1 = m_{k-1} G'_\perp$.*

One consequence of this theorem is that we can replace any optimal G with an equivalent G' in which the first local coin has exactly two outcomes, and in which the adversary never prefers hiding a local coin to revealing one. Since the theorem also applies recursively to all subgames of G, we may assume that these conditions in fact hold throughout G'. Thus no additional power is obtained by allowing more than two outcomes to a coin. However, the theorem does not imply that we can require that all local coins are fair; indeed, for most optimal games they will not be.

In addition, we have shown the following about the shape of the curves corresponding to optimal games:

Theorem 2 *Fix n and k with $k < n$. For each x in $[-1,1]$, let $f(x)$ be the smallest value of $M_k G$ for all G such that $m_k G = x$. Then f is nondecreasing and concave.*

Unfortunately, with the exception of some extreme cases like $k = n-1$, the (n,k) curves do not appear to have nice algebraic descriptions. So while in principle equations (1) and (2) and the minimum-of-hypotenuses construction constrain the curves completely, to obtain any useful bounds from them we will be forced to resort to approximation.

2.2 Lower Bounds for Fixed-Length Games

The essential idea of our lower bound for fixed-length coin-flipping games is to choose a family of functions to act as lower bounds for the optimal curves as defined above, and show by repeating the inscribed-right-triangle argument with these functions that they do in fact provide lower bounds on the optimal curves given appropriate parameters. The particular family of functions that we use consists of all hyperbolas that are symmetric about the diagonal from $(-1,1)$ to $(1,-1)$ and that pass through the corner points $(-1,-1)$ and $(1,1)$.[3] These hyperbolas are conveniently given by

$$\tanh^{-1} y - \tanh^{-1} x = c$$

[3]We conjecture (Conjecture 20) that a slightly tighter lower bound could be proven using the curves given by $\Phi^{-1}(y) - \Phi^{-1}(x) = c$, where Φ is the normal distribution function. An analog of Theorem 3 using Φ instead of tanh would improve the consensus lower bound in Theorem 19 by a logarithmic factor.

for various values of c. The linear $(n, 0)$ curve corresponds exactly to $c = 0$; the (n, n) curve is the limit as c goes to infinity. Our goal is to compute values of c as a function of n and k such that for all length-n games,

$$\tanh^{-1} M_k G - \tanh^{-1} m_k G \geq c(n, k).$$

Given $c(n-1, k)$ and $c(n-1, k-1)$, repeating the inscribed-right-triangle construction for the resulting hyperbolas is a not very difficult exercise in analytic geometry. Unfortunately, finding the particular point on the hypotenuse of the particular triangle that minimizes $c(n, k)$ is a bit more involved (details of both steps are given in the next two sections). The ultimate result of these efforts is:

Theorem 3 *Let G be a game of length n with outcome set $\{-1, +1\}$. Then for any $k \geq 0$, either $M_k G = 1$, $m_k G = -1$, or*

$$\tanh^{-1} M_k G - \tanh^{-1} m_k G \geq \frac{k}{2\sqrt{n}}. \tag{3}$$

2.2.1 Proof of Theorem 3

In this section we assume that each game has length n in all executions. Our results about such games also apply to any game whose *maximum* length is n, since we can always extend a branch that terminates early with dummy coin-flips that do not affect the outcome.

The proof is by induction on n. The case $n = 0$ is trivial. For $n = 1$, we have either $k = 0$, in which case $M_k G = m_k G$ and both sides of (3) are zero, or $k \geq 1$, and either $G_\perp = 1$ and thus $M_k G = G_\perp = 1$ or $G_\perp = -1$ and thus $m_k G = G_\perp = -1$.

For larger values of n, we wish to show that if the inequality holds for n it holds for $n + 1$. Observe first that if $k = 0$ we again have $M_k G = m_k G$ and the theorem holds. Thus it remains only to consider the case $k > 0$.

Suppose that the inequality holds for length n games and consider a length $n + 1$ game G. Consider the pair $(m_k G, M_k G)$ as a point in $[-1, 1]^2$. The coordinates of this point are averages over the same distribution; thus we can treat the point itself as an average (as a two-dimensional vector) of a set S of points in $[-1, 1]^2$. The coordinates of the points in this set are given by $(\min(m_k G_\alpha, m_{k-1} G_\perp), \max(M_k G_\alpha, M_{k-1} G_\perp))$ for each possible value of α.

Each point (x, y) in S must satisfy three constraints: (i) x is at least $m_{k-1} G_\perp$; (ii) y is at most $M_{k-1} G_\perp$; and (iii) $\tanh^{-1} y - \tanh^{-1} x \geq \frac{k}{2\sqrt{n}}$ (by applying the induction hypothesis to G_α). The region R defined by these three constraints looks like a rectangle with a concave bite taken out of its bottom right corner, which is the corner with coordinates $(m_{k-1} G_\perp, M_{k-1} G_\perp)$. What is useful about this region is that it is defined solely in terms of n, k, and the choice of $G_\perp$; and we know that any length n game G has payoffs $(m_k G, M_k G)$ that, as averages of points in the region, must lie somewhere in its convex closure $\overline{R}$.

Thus we can prove that our inequality holds for all games G by proving that it holds for any point in the convex closure of a region defined as above.

Let's start with the choice of $G_\perp$. By the induction hypothesis,

$$\tanh^{-1} M_{k-1}G_\perp - \tanh^{-1} m_{k-1}G_\perp \geq \frac{k-1}{2\sqrt{n}}.$$

Thus there exists a z such that

$$M_{k-1}G_\perp \geq \tanh(z + \frac{k-1}{4\sqrt{n}})$$

and

$$m_{k-1}G_\perp \leq \tanh(z - \frac{k-1}{4\sqrt{n}}).$$

For the rest of the proof we will ignore the actual payoffs of $G_\perp$ and use instead the bounds $\tanh(z \pm \frac{k-1}{4\sqrt{n}})$.

Now let us consider the extreme points (x, y) on the curve $\tanh^{-1} y - \tanh^{-1} x = \frac{k}{2\sqrt{n}}$. When $y = z + \frac{k-1}{4\sqrt{n}}$, we have the point

$$(x_0, y_0) = (\tanh(z - \frac{k+1}{4\sqrt{n}}), \tanh(z + \frac{k-1}{4\sqrt{n}})).$$

When $x = z + \frac{k-1}{4\sqrt{n}}$, we get

$$(x_1, y_1) = (\tanh(z - \frac{k-1}{4\sqrt{n}}), \tanh(z + \frac{k+1}{4\sqrt{n}})).$$

We wish to show that every point in $\overline{R}$ is dominated by a convex combination of these two points.

Fix α and let $x = \min(m_k G_\alpha, m_{k-1}G_\perp)$ and $y = \max(M_k G_\alpha, M_{k-1}G_\perp)$. Define:

$$\lambda_\alpha = \begin{cases} 1 & \text{if } x \leq x_0, \\ \frac{x_1 - x}{x_1 - x_0} & \text{if } x_0 \leq x \leq x_1, \text{ and} \\ 0 & \text{if } x_1 \leq x. \end{cases}$$

Let $(x', y') = \lambda_\alpha(x_0, y_0) + (1 - \lambda_\alpha)(x_1, y_1)$. We claim that $x \leq x'$ and $y' \leq y$.

To prove this claim consider the three cases in the definition of λ_α separately. If $x \leq x_0$, then $x \leq x' = x_0$; furthermore $y' = y_0 \leq M_{k-1}G_\perp \leq \max(M_k G_\alpha, M_{k-1}G_\perp) = y$. A similar argument proves the claim when $\lambda_\alpha = 0$. For the middle case, we have $x = x' = \lambda_\alpha x_0 + (1 - \lambda_\alpha x_1)$ and $y \geq \tanh\left(\tanh^{-1}(x) + \frac{k}{2\sqrt{n}}\right)$ which is at least $\lambda_\alpha y_0 + (1 - \lambda_\alpha)y_1$ by Lemma 7. Thus the claim holds.

Let $\lambda = \mathrm{E}_\alpha[\lambda_\alpha]$. From the claim it follows that

$$\begin{aligned} m_k G &= \mathrm{E}_\alpha\left[\min(m_k G_\alpha, m_{k-1}G_\perp)\right] \\ &\leq \mathrm{E}_\alpha\left[\lambda_\alpha x_0 + (1 - \lambda_\alpha)x_1\right] \\ &= \mathrm{E}_\alpha[\lambda_\alpha]x_0 + (1 - \mathrm{E}_\alpha[\lambda_\alpha])x_1 \\ &= \lambda \tanh\left(z - \frac{k+1}{4\sqrt{n}}\right) + (1 - \lambda)\tanh\left(z - \frac{k-1}{4\sqrt{n}}\right). \end{aligned}$$

Similarly we have

$$\begin{aligned} M_k G &= \mathrm{E}_\alpha\left[\max(M_k G_\alpha, M_{k-1} G_\perp)\right] \\ &\geq \mathrm{E}_\alpha\left[\lambda_\alpha y_0 + (1-\lambda_\alpha) y_1\right] \\ &= \mathrm{E}_\alpha[\lambda_\alpha] y_0 + (1 - \mathrm{E}_\alpha[\lambda_\alpha]) y_1 \\ &= \lambda \tanh\left(z + \frac{k-1}{4\sqrt{n}}\right) + (1-\lambda)\tanh\left(z + \frac{k+1}{4\sqrt{n}}\right). \end{aligned}$$

We are left with the task of reducing this expression to a more convenient form. To do so we apply several inequalities involving hyperbolic functions, proved in the next section. In particular the second-to-last inequality below is given by Lemma 5 and the last is given by Lemma 6.

$$\begin{aligned} &\tanh^{-1} M_k G - \tanh^{-1} m_k G \\ &\geq \tanh^{-1}\left[\lambda \tanh\left(z + \frac{k-1}{4\sqrt{n}}\right) + (1-\lambda)\tanh\left(z + \frac{k+1}{4\sqrt{n}}\right)\right] \\ &\quad - \tanh^{-1}\left[\lambda \tanh\left(z - \frac{k+1}{4\sqrt{n}}\right) + (1-\lambda)\tanh\left(z - \frac{k-1}{4\sqrt{n}}\right)\right] \\ &\geq 2\tanh^{-1}\left[\frac{1}{2}\tanh\frac{k-1}{4\sqrt{n}} + \frac{1}{2}\tanh\frac{k+1}{4\sqrt{n}}\right] \\ &\geq 2\tanh^{-1}\tanh\left(\frac{k}{4\sqrt{n}}\operatorname{sech}^2\left(\frac{1}{2\sqrt{n}}\right)\right) \\ &= \frac{k}{2\sqrt{n}}\operatorname{sech}^2\left(\frac{1}{2\sqrt{n}}\right). \end{aligned}$$

It remains only to show for all $k \geq 1$ and $n \geq 1$ that

$$\frac{k}{2\sqrt{n}}\operatorname{sech}^2\left(\frac{1}{2\sqrt{n}}\right) \geq \frac{k}{2\sqrt{n+1}}.$$

From the Taylor's series expansion of sech z we have $\operatorname{sech} z \geq 1 - \frac{1}{2}z^2$. Setting $z = \frac{1}{2\sqrt{n}}$ gives $\operatorname{sech}\frac{1}{2\sqrt{n}} \geq 1 - \frac{1}{8n}$. But then $\operatorname{sech}^2\frac{1}{2\sqrt{n}} \geq 1 - \frac{1}{4n}$ and $\operatorname{sech}^4\frac{1}{2\sqrt{n}} \geq 1 - \frac{1}{2n}$. Now for $n \geq 1$, $1 - \frac{1}{2n} \geq 1 - \frac{1}{n+1} = \frac{n}{n+1}$. Thus we have $\operatorname{sech}^4\frac{1}{2\sqrt{n}} \geq \frac{n}{n+1}$ so $\operatorname{sech}^2\frac{1}{2\sqrt{n}} \geq \sqrt{\frac{n}{n+1}}$ and $\frac{k}{2\sqrt{n}}\operatorname{sech}^2\left(\frac{1}{2\sqrt{n}}\right) \geq \frac{k}{2\sqrt{n+1}}$.

Thus if G is a length $n+1$ game, we have that

$$\tanh^{-1} M_k G - \tanh^{-1} m_k G \geq \frac{k}{2\sqrt{n+1}}$$

and the induction goes through.

2.2.2 Some Inequalities Involving Hyperbolic Functions

These are used in the proof of Theorem 3.

Lemma 4 *Let $0 \le A \le B < 1$. Then*

$$(1+A)(1+B)(2-A-B)^2 \ge (1-A)(1-B)(2+A+B)^2. \tag{4}$$

Proof: Each of the inequalities below is implied by the one that follows it:

$$\begin{aligned}
(1+A)(1+B)(2-A-B)^2 &\ge (1-A)(1-B)(2+A+B)^2 \\
\frac{(2-A-B)^2}{(1-A)(1-B)} &\ge \frac{(2+A+B)^2}{(1+A)(1+B)} \\
\frac{4-4A-4B+A^2+2AB+B^2}{1-A-B+AB} &\ge \frac{4+4A+4B+A^2+2AB+B^2}{1+A+B+AB} \\
4+\frac{A^2-2AB+B^2}{1-A-B+AB} &\ge 4+\frac{A^2-2AB+B^2}{1+A+B+AB} \\
\frac{(A-B)^2}{1-A-B+AB} &\ge \frac{(A-B)^2}{1+A+B+AB} \\
\frac{1}{1-A-B+AB} &\ge \frac{1}{1+A+B+AB} \\
1+A+B+AB &\ge 1-A-B+AB \\
A+B &\ge -A-B
\end{aligned}$$

and this last inequality follows from $A+B \ge 0$. ■

Lemma 5 *Let $0 \le a \le b$. Then for all x and all λ such that $0 \le \lambda \le 1$,*

$$\begin{aligned}
&\tanh^{-1}\left(\lambda \tanh(x+a) + (1-\lambda)\tanh(x+b)\right) \\
&-\tanh^{-1}\left(\lambda \tanh(x-b) + (1-\lambda)\tanh(x-a)\right) \\
\ge\ & 2\tanh^{-1}\left(\frac{1}{2}\tanh a + \frac{1}{2}\tanh b\right).
\end{aligned} \tag{5}$$

Proof: Equality holds when $a = b$ or $\lambda = \frac{1}{2}$ and $x = 0$, so we can prove the inequality in general by showing that for fixed a and b with $a < b$ the left-hand side L of (5) is minimized when $\lambda = \frac{1}{2}$ and $x = 0$.

To do so we will take L through a sequence of transformations resulting in a rational function in λ, $\tanh a$, $\tanh b$, and $\tanh x$. Showing that this function is minimized when $\lambda = \frac{1}{2}$ and $x = 0$ is equivalent to showing that a certain polynomial obtained by multiplying out denominators is never negative. This problem can in turned be reduced to showing that the polynomial is never negative for certain extreme cases, where its sign can easily be determined. Reversing these steps proves the original bound.

Step 1: Removing occurrences of $\tanh^{-1}$ from L. The first step is to remove the inverse hyperbolic tangents that appear in L. To save space let us write $\overline{\lambda}$ for $1-\lambda$, yielding

$$\begin{aligned}
L \;=\; & \tanh^{-1}\left(\lambda \tanh(x+a) + \overline{\lambda}\tanh(x+b)\right) \\
& -\tanh^{-1}\left(\lambda \tanh(x-b) + \overline{\lambda}\tanh(x-a)\right)
\end{aligned}$$

This we can rewrite using the fact tanh and $\tanh^{-1}$ are both odd functions to get:

$$\begin{aligned} L &= \tanh^{-1}\left(\lambda \tanh(a+x) + \overline{\lambda}\tanh(b+x)\right) \\ &\quad + \tanh^{-1}\left(\lambda \tanh(b-x) + \overline{\lambda}\tanh(a-x)\right) \end{aligned}$$

Let $s = \lambda \tanh(a+x) + \overline{\lambda}\tanh(b+x)$ and let $t = \lambda \tanh(b-x) + \overline{\lambda}\tanh(a-x)$. Recall that for $|z| < 1$, $\tanh^{-1} z = \frac{1}{2}\ln\frac{1+x}{1-x}$. Thus

$$L = \tanh^{-1} s + \tanh^{-1} t = \frac{1}{2}\ln\frac{1+s}{1-s} + \frac{1}{2}\ln\frac{1+t}{1-t} = \frac{1}{2}\ln\frac{(1+s)(1+t)}{(1-s)(1-t)}.$$

Thus to minimize L we need to minimize $\frac{(1+s)(1+t)}{(1-s)(1-t)}$.

Step 2: Further expansion using the sum formula for tanh. To do so we will first expand every occurrence of tanh in s and t using the identity $\tanh(x \pm y) = (\tanh x \pm \tanh y)/(1 \pm \tanh x \tanh y)$. In order to give the resulting expressions even the slightest hope of readability, let us write X for $\tanh x$, A for $\tanh a$, and B for $\tanh b$. We have

$$s = \lambda \tanh(a+x) + \overline{\lambda}\tanh(b+x) = \lambda\frac{A+X}{1+AX} + \overline{\lambda}\frac{B+X}{1+BX}.$$

Thus

$$\begin{aligned} 1+s &= 1 + \lambda\frac{A+X}{1+AX} + \overline{\lambda}\frac{B+X}{1+BX} \\ &= \lambda\frac{1+AX+A+X}{1+AX} + \overline{\lambda}\frac{1+BX+B+X}{1+BX} \\ &= \lambda\frac{(1+A)(1+X)}{1+AX} + \overline{\lambda}\frac{(1+B)(1+X)}{1+BX}. \end{aligned}$$

A similar expansion shows that

$$\begin{aligned} 1-s &= \lambda\frac{(1-A)(1-X)}{1+AX} + \overline{\lambda}\frac{(1-B)(1-X)}{1+BX}, \\ 1+t &= \lambda\frac{(1+B)(1-X)}{1-BX} + \overline{\lambda}\frac{(1+A)(1-X)}{1-AX}, \text{ and} \\ 1-t &= \lambda\frac{(1-B)(1+X)}{1-BX} + \overline{\lambda}\frac{(1-A)(1+X)}{1-AX}; \end{aligned}$$

from which it follows that

$$\begin{aligned} &\frac{(1+s)(1+t)}{(1-s)(1-t)} \\ &= \frac{\left(\lambda\frac{(1+A)(1+X)}{1+AX} + \overline{\lambda}\frac{(1+B)(1+X)}{1+BX}\right)\left(\lambda\frac{(1+B)(1-X)}{1-BX} + \overline{\lambda}\frac{(1+A)(1-X)}{1-AX}\right)}{\left(\lambda\frac{(1-A)(1-X)}{1+AX} + \overline{\lambda}\frac{(1-B)(1-X)}{1+BX}\right)\left(\lambda\frac{(1-B)(1+X)}{1-BX} + \overline{\lambda}\frac{(1-A)(1+X)}{1-AX}\right)} \end{aligned}$$

$$
\begin{aligned}
&= \frac{\left(\lambda\frac{1+A}{1+AX}+\overline{\lambda}\frac{1+B}{1+BX}\right)\left(\lambda\frac{1+B}{1-BX}+\overline{\lambda}\frac{1+A}{1-AX}\right)}{\left(\lambda\frac{1-A}{1+AX}+\overline{\lambda}\frac{1-B}{1+BX}\right)\left(\lambda\frac{1-B}{1-BX}+\overline{\lambda}\frac{1-A}{1-AX}\right)} \\
&= \frac{\begin{array}{l}\left[\lambda(1+A)(1+BX)+\overline{\lambda}(1+B)(1+AX)\right] \\ \cdot\left[\lambda(1+B)(1-AX)+\overline{\lambda}(1+A)(1-BX)\right]\end{array}}{\begin{array}{l}\left[\lambda(1-A)(1+BX)+\overline{\lambda}(1-B)(1+AX)\right] \\ \cdot\left[\lambda(1-B)(1-AX)+\overline{\lambda}(1-A)(1-BX)\right]\end{array}} \qquad (6)
\end{aligned}
$$

Step 3: Transforming a rational function inequality to a polynomial inequality. The next step is to reduce the problem of showing that the rational function (6) is minimized at $x = 0$, $\lambda = \frac{1}{2}$ to an inequality involving only polynomials.

This transformation will be less cumbersome if we can find a way to write (6) more compactly. We've already canceled all the terms that cancel easily; so to simplify it further we are going to need to exploit its internal symmetry. Let $\Delta = 2\lambda - 1$, so that $\lambda = \frac{1+\Delta}{2}$ and $\overline{\lambda} = 1 - \lambda = \frac{1-\Delta}{2}$. Let

$$
\begin{aligned}
R &= (1+\Delta)(1+A)(1+BX) + (1-\Delta)(1+B)(1+AX) \\
S &= (1-\Delta)(1+A)(1-BX) + (1+\Delta)(1+B)(1-AX) \\
T &= (1+\Delta)(1-A)(1+BX) + (1-\Delta)(1-B)(1+AX) \\
U &= (1-\Delta)(1-A)(1-BX) + (1+\Delta)(1-B)(1-AX)
\end{aligned}
$$

So that (6) is $\frac{RS}{TU}$ (we are canceling out a few factors of 2 here).

Let us now consider what happens if we set $X = 0$ and $\Delta = 0$ (i.e., $x = 0$ and $\lambda = \overline{\lambda} = \frac{1}{2}$). Then R, S, T, and U are all radically simplified and (6) becomes $\frac{P^2}{Q^2}$ where $P = 2 + A + B$ and $Q = 2 - A - B$. Since our goal is to show that $\frac{RS}{TU}$ is minimized at $X = 0, \Delta = 0$, we must demonstrate that for any values for X and Δ with $|X| < 1$ and $|\Delta| \leq 1$,

$$
\frac{RS}{TU} \geq \frac{P^2}{Q^2} \qquad (7)
$$

Observe that since $|\tanh z| < 1$ for all z, we have $|A| < 1$, $|B| < 1$, and $|X| < 1$. It follows that both T and U are positive and thus the inequality (7) holds just in case $RSQ^2 \geq TUP^2$ or $RSQ^2 - TUP^2 \geq 0$.

Step 4: Constraining the coefficients of the polynomial. Consider $f(\Delta, X) = RSQ^2 - TUP^2$ as a polynomial in Δ and X. If possible, we'd like to show f is always non-negative without having to multiply out its many terms. Fortunately, we can get quite a bit of information about its coefficients without such Herculean efforts.

Let a_{ij} be the coefficient in f of $\Delta^i X^j$. If $\Delta = X = 0$, then $RSQ^2 - TUP^2 = P^2Q^2 - Q^2P^2 = 0$. Thus $a_{00} = 0$. By symmetry $f(\Delta, X) = f(-\Delta, -X)$ (changing both of these signs swaps R with S and T with U). Thus $a_{ij} = 0$ for

any i, j such that $i+j$ is odd. Finally, since the largest power of Δ or X in each of R, S, T, and U is 1, and neither Δ nor X appears in P or Q, we have that $a_{ij}=0$ whenever i or j is greater than 2. This leaves four possible nonzero coefficients, and so we can write f as $a_{11}\Delta X + a_{20}\Delta^2 + a_{02}X^2 + a_{22}\Delta^2X^2$.

Step 5: Reduction to extreme cases. Now we wish to show that if f is negative anywhere in $[-1,1]^2$, it is negative for some point (Δ, X) with either $\Delta = 1$ or $X = 1$. To do so we will show that any (Δ, X) in the interior of $[-1,1]^2$ that yields a negative f can be replaced by $t\Delta, tX$ for any t such that $|t|>1$, with the result that $f(t\Delta, tX)$ will also give a negative f. If all of the terms in f had the same degree, this would be easy; since this is not the case, we must first show that the coefficient a_{22} of the Δ^2X^2 term is negative.

Fortunately, with not too much work a_{22} is seen to be $(B-A)(B-A)Q^2 - (B-A)(B-A)P^2$ or $(B-A)^2[(2-(A+B))^2-(2+(A+B))^2] = (B-A)^2[-8(A+B)^2] \le 0$. So if $|t|>1$,

$$\begin{aligned} f(t\Delta, tX) &= a_{11}t^2\Delta X + a_{20}t^2\Delta^2 + a_{02}t^2X^2 + a_{22}t^4\Delta^2X^2 \\ &= t^2(a_{11}\Delta X a_{20}\Delta^2 + a_{02}X^2 + a_{22}t^2\Delta^2X^2) \\ &< t^2(a_{11}\Delta X a_{20}\Delta^2 + a_{02}X^2 + a_{22}\Delta^2X^2) \\ &= t^2 f(\Delta, X). \end{aligned} \tag{8}$$

Thus if f is ever negative on $[-1,1]^2$, it is negative for some point in which $\Delta = 1$ or $X = 1$, since we can choose whichever of Δ or X has larger absolute magnitude and set $t = 1/\Delta$ or $t = 1/X$.

Step 5a: $\Delta = 1$, $X \neq 1$. Let us examine the $\Delta = 1$ case first. We will assume that $|X|<1$; the case $\Delta = 1$, $X = 1$ will be covered by the $X = 1$ case below. Recall that f is negative if and only if the inequality (5) is violated. If $\Delta = 1$, then $\lambda = 1$ and (5) becomes

$$\tanh^{-1}\tanh(x+a) - \tanh^{-1}\tanh(x-b) \ge 2\tanh^{-1}\left(\frac{1}{2}\tanh a + \frac{1}{2}\tanh b\right). \tag{9}$$

The left-hand side of this inequality simplifies to $a+b$, and so (9) holds just in case

$$\tanh\frac{a+b}{2} \ge \frac{1}{2}\tanh a + \frac{1}{2}\tanh b,$$

which holds because tanh is concave on the positive real line.

Step 5b: $X = 1$. When $X = 1$, we have

$$\begin{aligned} R &= (1+\Delta)(1+A)(1+B) + (1-\Delta)(1+A)(1+B) \\ &= 2(1+A)(1+B) \\ S &= (1-\Delta)(1+A)(1-B) + (1+\Delta)(1-A)(1+B) \\ T &= (1+\Delta)(1-A)(1+B) + (1-\Delta)(1+A)(1-B) \end{aligned}$$

$$\begin{aligned} & = & S \\ U & = & (1-\Delta)(1-A)(1-B) + (1+\Delta)(1-A)(1-B) \\ & = & 2(1-A)(1-B). \end{aligned}$$

Thus $RSQ^2 - TUP^2 = 2S(1+A)(1+B)(2-A-B)^2 - 2S(1-A)(1-B)(2+A+B)^2$ which is non-negative by Lemma 4.

Wrap-up. In summary, we have that $f(\Delta, X) \geq 0$ whenever $\Delta = 1$ or $X = 1$. Using (8), this implies that $f(\Delta, X) \geq 0$ for all points (Δ, X) in the unit square $[-1, 1]^2$, which, after reversing the translations from (5) to f, implies that (5) holds under the conditions stated in the Lemma. ■

Lemma 6 *If $x \geq 0$, then for any a,*

$$\tanh(x+a) + \tanh(x-a) \geq 2\tanh(x \operatorname{sech}^2 2a). \tag{10}$$

Proof: The inequality above holds just in case

$$\tanh(x+a) + \tanh(x-a) - 2\tanh(x \operatorname{sech}^2 2a) \tag{11}$$

is non-negative for non-negative x. To avoid unwieldy notation, let us write c for $\operatorname{sech}^2 2a$. Observe that $\tanh x = \frac{e^{2x}-1}{e^{2x}+1} = 1 - \frac{2}{e^{2x}+1}$. So we can rewrite $\tanh(x+a) + tanh(x-a) - 2\tanh(xc)$ as

$$\begin{aligned} & \left(1 - \frac{2}{e^{2x+2a}+1}\right) + \left(1 - \frac{2}{e^{2x-2a}+1}\right) - \left(2 + 2\frac{2}{e^{2cx}+1}\right) \\ = & \frac{4}{e^{2cx}+1} - \frac{2}{e^{2x+2a}+1} - \frac{2}{e^{2x-2a}+1}. \end{aligned}$$

Note that each of these denominators is positive. Thus multiplying out the denominators and dividing by 2 does not change the sign, and the sign of the original expression is the same as the sign of

$$\begin{aligned} & 2(e^{2x+2a}+1)(e^{2x-2a}+1) \\ & -(e^{2cx}+1)(e^{2x-2a}+1) \\ & -(e^{2cx}+1)(e^{2x+2a}+1) \\ = & 2(e^{4x} + e^{2x+2a} + e^{2x-2a} + 1) \\ & -(e^{2cx}e^{2x-2a} + e^{2x-2a} + e^{2cx} + 1) \\ & -(e^{2cx}e^{2x+2a} + e^{2x+2a} + e^{2cx} + 1) \\ = & 2e^{4x} + e^{2x+2a} + e^{2x-2a} - e^{2cx}\left(e^{2x+2a} + e^{2x-2a}\right) - 2e^{2cx} \\ = & e^{2x}\left[2e^{2x} - 2e^{2(c-1)x} + \left(1 - e^{2cx}\right)\left(e^{2a} + e^{-2a}\right)\right]. \end{aligned}$$

Since $e^{2x} > 0$, we can drop the first factor while preserving the sign. Writing z for e^{2x}, and noting that $e^{2a} + e^{-2a} = 2\cosh 2a$, the second factor becomes

$$2z - 2z^{c-1} + 2(1 - z^c)\cosh 2a,$$

and now we can divide out 2 without changing the sign to obtain

$$z - z^{c-1} + (1 - z^c)\cosh 2a. \tag{12}$$

To show that the inequality (10) holds when $x \geq 0$, it is necessary to show that the sign of (12), and thus of (11), is non-negative for $z \geq 1$. Note that when $x = 0$, $z = e^{2x} = 1$ and (12) reduces to 0. So if we can show that (12) is non-decreasing for $z \geq 1$ we are done.

This we do by taking the derivative of (12) with respect to z and showing that it is non-negative when $z \geq 1$. The derivative is $1 - (c-1)z^{c-2} - cz^{c-1}\cosh 2a$. Observe that $c = \operatorname{sech}^2 2a \leq 1$ and $z \geq 1$ implies both $z^{c-1} \leq 1$ and $z^{c-2} \leq 1$. Thus we have

$$\begin{aligned}
& 1 - (c-1)z^{c-2} - cz^{c-1}\cosh 2a \\
\geq\ & 1 - (c-1) - c\cosh 2a \\
=\ & 1 - (\operatorname{sech}^2 2a - 1) - \operatorname{sech}^2 2a \cosh 2a \\
=\ & 2 - \operatorname{sech}^2 2a - \operatorname{sech} 2a \\
\geq\ & 0.
\end{aligned}$$

■

Lemma 7 *If $a \geq 0$, then the function $f(x) = \tanh(a + \tanh^{-1} x)$ is monotone increasing and concave.*

Proof: That f is monotone follows immediately from the monotonicity of tanh and $\tanh^{-1}$. To show it is concave, observe that:

$$\begin{aligned}
& \frac{d^2}{dx^2}\tanh(a + \tanh^{-1} x) \\
=\ & \frac{d}{dx}\operatorname{sech}^2(a + \tanh^{-1} x)\frac{1}{1-x^2} \\
=\ & 2\operatorname{sech}(a + \tanh^{-1} x)[-\operatorname{sech}(a + \tanh^{-1} x)\tanh(a + \tanh^{-1} x)]\frac{1}{(1-x^2)^2} \\
& + \operatorname{sech}^2(a + \tanh^{-1} x)\frac{-1}{(1-x^2)^2}2x \\
=\ & -2\operatorname{sech}^2(a + \tanh^{-1} x)\frac{1}{(1-x^2)^2}\left[\tanh(a + \tanh^{-1} x) - x\right] \\
\leq\ & 0.
\end{aligned}$$

Note that in the last step we need the fact that f is monotone to know that the last factor is positive. ■

2.2.3 Corollaries to Theorem 3

Theorem 3 assumes a coin-flipping game with ± 1 outcomes. For more general sets of outcomes it is more convenient to work with the minimum and maximum probabilities of some particular outcome rather than the expected outcome. A simple transformation of the theorem gives:

Corollary 8 *Let G be a coin-flipping game and let x lie in the outcome set of G. Fix k, and let $p = \min_A \Pr[G \circ A = x]$ and $q = \min_A \Pr[G \circ A \neq x]$, where in each case A ranges over adversaries that can hide up to k local coins. Then*

$$\ln \frac{1-p}{p} + \ln \frac{1-q}{q} \geq \frac{k}{2\sqrt{n}}. \tag{13}$$

Proof: Consider the modification G' of G which replaces each x outcome with $+1$ and each non-x outcome with -1. Then $m_k G = \min_A \mathrm{E}[G \circ A] = p - (1-p) = 2p - 1$ and $M_k G = \max_A \mathrm{E}[G \circ A] = (1-q) - q = 1 - 2q$. Thus $\tanh^{-1}(M_k G) = \tanh^{-1}(1-2q) = \ln \frac{1+(1-2q)}{1-(1-2q)} = \ln \frac{2-2q}{2q} = \ln \frac{1-q}{q}$ and $-\tanh^{-1}(m_k G) = -\tanh^{-1}(2p-1) = \tanh^{-1}(1-2p) = \ln \frac{1-p}{p}$.

Substituting into (3) in Theorem 3 then gives the desired result. ■

If the bound on each side is the same, we can simplify even further:

Corollary 9 *Let G be a coin-flipping game, let x be one of its outcomes, and let A range over adversaries that can hide up to k local coins. If for some $\epsilon < \frac{1}{2}$, $\min_A \Pr[G \circ A = x] \geq \epsilon$ and $\min_A \Pr[G \circ A \neq x] \geq \epsilon$, then the maximum length n of G is at least*

$$\frac{k^2}{16 \ln^2 \left(\frac{1}{\epsilon} - 1\right)}$$

Proof: From the previous corollary we have that $\frac{k}{2\sqrt{n}} \geq 2 \ln \frac{1-\epsilon}{\epsilon} = 2 \ln \left(\frac{1}{\epsilon} - 1\right)$. Since $\epsilon < \frac{1}{2}$, the logarithmic term is positive and we can rearrange this inequality to get the desired bound. ■

2.3 Lower Bounds for Variable-Length Games

In the preceding section we considered the connection between the adversary's influence over the outcome and the *maximum* length of a game. Here we consider instead the connection between the adversary's influence and the worst-case *expected* length of a game. In principle one could imagine low-expected-length games whose small bias was purchased by a high maximum length in rare executions; thus the bounds on maximum length do not immediately imply bounds on expected length.

However, using a truncation argument, we can show that a bound similar to that given in Corollary 9 holds even if we are considering the *expected* length of G rather than its maximum length. The theorem below covers both the worst-case expected length (when the adversary is trying to maximize the running time of the protocol) and the best-case expected length (when the adversary is trying to minimize the running time of the protocol). The worst-case bound will be used later to get a lower bound on the work required for consensus.

Theorem 10 *Fix k, and let A range over adversaries that can hide up to k local coins. Let G be a coin-flipping game with an outcome x such that* $\min_A \Pr[G \circ$

$A = x] \geq \epsilon$ *and* $\min_A \Pr[G \circ A \neq x] \geq \epsilon$. *Then the worst-case expected length of* G *is at least*

$$\frac{3}{64} \cdot \frac{k^2}{\ln^2\left(\frac{1}{\epsilon/2} - 1\right)}$$

and the best-case expected length is at least

$$\frac{1}{32} \cdot \frac{\epsilon k^2}{\ln^2\left(\frac{1}{\epsilon/2} - 1\right)}.$$

Proof: The essential method is to show that if a game exists whose expected length is "too good" then a truncated version of this game exists that violates the requirements of Corollary 9.

Let us assume without loss of generality that G has outcomes $x = 1$ and 0. (We can justify this assumption by replacing all x outcomes with 1 and all non-x outcomes with 0.)

First, the worst-case bound. Let

$$m = \frac{k^2}{16 \ln^2\left(\frac{1}{\epsilon/2} - 1\right)}.$$

Let G_m be the game obtained by truncating G as follows. If G finishes in m or fewer steps, let $G_m = G$. If G finishes in more than m steps, let $G_m = \perp$. The value of m is chosen such that for any outcome x of G_m, at least one of $\min_A \Pr[G \circ A = x]$ and $\min_A \Pr[G \circ A \neq x]$ is less than or equal to $\epsilon/2$ (using Corollary 9).

For each A and each execution of $G \circ A$ that produces an outcome v, there is an execution of $G_m \circ A$ that produces either v or $\perp$. It is given that $\min_A \Pr[G \circ A = 0]$ is at least ϵ; thus it follows that $\epsilon \leq \min_A \Pr[G_m \circ A \in \{0, \perp\}] = \min_A \Pr[G_m \circ A \neq 1]$. But then $\min_A \Pr[G_m \circ A = 1] < \epsilon/2$. Since for any A, $\Pr[G_m \circ A \in \{1, \perp\}] \geq \epsilon$, we get $\min_A \Pr[G_m \circ A = \perp] > \epsilon/2$. But applying Corollary 9 a second time now implies that $\min_A \Pr[G_m \circ A \neq \perp] \leq \epsilon/2$; in other words, that for some adversary A the probability that $G \circ A$ does not finish after m steps is at least $1 - \epsilon/2$. Since $\epsilon < 1/2$, this probability is at least $3/4$ and so the worst-case expected length of G is at least $(3/4)m$.

The best-case bound is also obtained by considering a truncated game. Let $T = \min_A \mathrm{E}[\mathrm{length}(G \circ A)]$. Let $n = \frac{2T}{\epsilon}$, so that (using Markov's inequality) the probability that $G \circ A$ has not finished by time n is at most $\epsilon/2$. Thus $\min_A \Pr[G_n \circ A = \perp] \leq \epsilon/2$. But $\min_A \Pr[G_n \circ A \in \{0, \perp\}] \geq \min_A \Pr[G \circ A = 0] \geq \epsilon$, so $\min_A \Pr[G_n \circ A = 0] \geq \epsilon/2$. By symmetry we also have $\min_A \Pr[G_n \circ A = 1] \geq \epsilon/2$. Corollary 9 then gives

$$n \geq \frac{1}{16} \cdot \frac{k^2}{\ln^2\left(\frac{1}{\epsilon/2} - 1\right)}$$

and thus

$$T = n \cdot \frac{\epsilon}{2} \geq \frac{1}{32} \cdot \frac{\epsilon k^2}{\ln^2\left(\frac{1}{\epsilon/2} - 1\right)}.$$

■

2.4 Consequences for Constant-Bias Coins

For constant bias, Corollary 9 and Theorem 10 imply that we need $\Omega(t^2)$ local coin flips in both the worst and average cases. This is true even though the adversary's power is limited by the fact that (a) the local coin flips may have arbitrary ranges and distributions; (b) the adversary can hide coins, but cannot control them; (c) the adversary must decide which coins to hide or reveal immediately in an on-line fashion; and (d) the algorithm may observe and react to the choices of which coins to hide. These assumptions were chosen to minimize the power of the adversary while still capturing the essence of its powers in a distributed system with failures.

In contrast, it is not difficult to see that taking a majority of $\Theta(t^2)$ fair coins gives a constant bias even if (a) local coins are required to be fair random bits; (b) the adversary can replace up to t values with new values of its own choosing; (c) the adversary may observe the values of all the local coins before deciding which ones to alter; and (d) changes made by the adversary are invisible to the algorithm. So the $\Omega(t^2)$ lower bound for constant bias is tight for a wide range of assumptions about the powers of the algorithm and the adversary.[4]

2.5 Connection to Randomized Distributed Algorithms with Failures

The importance of coin-flipping games as defined above comes from the fact that they can often be found embedded inside randomized distributed algorithms. Let us discuss briefly how this embedding works.

Consider a randomized distributed algorithm in a model in which (a) all random events are internal to individual processes; and (b) all other nondeterminism is under the control of an adaptive adversary. Suppose further that the adversary has the power to kill up to k of the processes. Then given any randomized algorithm in which some event X that does not depend on the states of faulty processes occurs with minimum probability m and maximum probability M, we can extract a coin-flipping game from it as follows. Arbitrarily fix all the nondeterministic choices of the adversary except for the decision whether

[4]The theorem does *not* apply if the adversary cannot observe local coin-flips, and so it cannot be used with an *oblivious* (as opposed to the usual *adaptive*) adversary. However, the bound on best-case expected length does imply that it is impossible to construct a "hybrid" constant-bias coin-flipping protocol that adapts to the strength of the adversary, finishing quickly against an oblivious adversary but using additional work to prevent an adaptive adversary from seizing control. This is not the case for consensus; for example, Chandra's consensus algorithm [Cha96] for a weak adversary switches over to an algorithm that is robust against an adaptive adversary if it does not finish in its usual time.

or not to kill each process immediately following each internal random event. (Since this step reduces the options of the adversary it can only increase m and decrease M.) Each step of the coin-flipping game corresponds to an execution of the distributed algorithm up to some such random event, which we interpret as the local coin. The adversary's choice to hide or reveal this local coin corresponds to its power to kill the process that executes the random event (thus preventing any other process from learning its value) or to let it run (which may or may not eventually reveal the value). The outcome of the coin-flipping game is determined by whether or not X occurs in the original system.

3 Lower Bound for Randomized Consensus

Consensus is a problem in which a group of n processes must agree on a bit. We will consider consensus in models in which at most t processes may fail by halting. Processes that do not halt (i.e., *correct* processes) must execute infinitely many operations. (A more detailed description of the model is given in Section 3.2.)

It is assumed that each process starts with some input bit and eventually *decides* on an output bit and then stops executing the algorithm. Formally, consensus is defined by three conditions:

- **Agreement.** All correct processes decide the same value with probability 1.
- **Non-triviality**. For each value v, there exists a set of inputs and an adversary that causes all correct processes to decide v with probability 1.
- **Termination.** All correct processes decide with probability 1.

Non-triviality is a rather weak condition, and for applications of consensus protocols a stronger condition is often more useful:

- **Validity.** If all processes have input v, all correct processes decide v with probability 1.

As non-triviality is implied by validity, if we show a lower bound on the total work of any protocol that satisfies agreement, non-triviality, and termination, we will have shown *a fortiori* a lower bound on any protocol that satisfies agreement, validity, and termination. Thus we will concentrate on consensus as defined by the first three conditions.

Since the agreement and termination conditions are violated only with probability zero, we can exclude all schedules in which they are violated without affecting the expected length of the protocol or the independence and unpredictability of local coin-flips. Thus without loss of generality we may assume that not only do agreement and termination apply to the protocol as a whole, but they also apply even if one conditions on starting with some particular finite execution α.

3.1 Overview of the Proof

In a randomized setting, we are concerned with the cost of carrying out a consensus protocol in terms of the expected total work when running against a worst-case adversary. We show how the coin-flipping lower bound can be used to show a lower bound on the worst-case expected cost of t-resilient randomized consensus in the standard asynchronous shared-memory model. As in the coin-flipping bound, we will measure the cost of a consensus protocol by the total number of local coin-flips executed by the processes. This measure is not affected by deterministic simulations, so any results we obtain for the shared-memory model will also apply to any model that can be simulated using shared memory, such as a t-resilient message-passing model.

For each adversary strategy and finite execution α there is a fixed probability that the protocol will decide 1 conditioned on the event that its execution starts with α. (We may speak without confusion of the protocol deciding 1, as opposed to individual processes deciding 1, because of the agreement condition.) For any set of adversaries, there is a range of probabilities running from the minimum to the maximum probability of deciding 1.

These ranges are used to define a probabilistic version of the bivalence and univalence conditions used in the well-known Fischer-Lynch-Paterson (FLP) impossibility proof for deterministic consensus [FLP85]. We will define an execution as *bivalent* if the adversary can force either outcome with high probability. A *v-valent* execution will be one after which only the outcome v can be forced with high probability. Finally, a *null-valent* execution will be one in which neither outcome can be forced with high probability. The notions of bivalence and v-valence (defined formally in Section 3.3) match the corresponding notions for deterministic algorithms used in the FLP proof; null-valence is new, as it cannot occur with a deterministic algorithm in which the probability of deciding each value v must always be exactly 0 or 1.

In outline, the proof that consensus is expensive for randomized algorithms retains much of the structure of the FLP proof. First, it is shown that with at least constant probability any protocol can be maneuvered from its initial state into either a bivalent or a null-valent execution. Once the protocol is in a bivalent execution, we show that there is a fair, failure-free extension that leads either to a local coin-flip or a null-valent execution. The result of flipping a local coin after a bivalent execution is, of course, random; but we can show that with high probability it leaves us with an execution which is either bivalent or null-valent or from which we are likely to return to a bivalent or a null-valent execution after additional coin-flips. If we do reach a null-valent execution, the coin-flipping bound applies.

Unlike a deterministic protocol, it is possible for a randomized protocol to "escape" through a local coin-flip into an execution in which it can finish the protocol quickly. But we will be able to show that the probability of escaping in this way is small, so that on average many local coin-flips will occur before it happens.

3.2 Model for Consensus Lower Bound

This section describes in detail the model used for the consensus lower bound. It is included for completeness, as lower bounds are notoriously sensitive to features of the underlying model. However, the reader who is familiar with previous work on asynchronous shared-memory systems will find no surprises here, and may wish to skip ahead to the actual proof starting in Section 3.3.

3.2.1 Foundations

There are many ways to represent a distributed system; we will use the I/O automaton model as described in [Lyn96]. In this model, an execution of a system is represented by a sequence $s_0, \pi_1, s_1, \pi_2, \ldots$ of alternating states and actions, starting with an initial state. An execution may be finite or infinite; if finite, it ends with a state. The behavior of a deterministic system is described by a transition relation consisting of triples (s_0, π, s_1) specifying the prior state, the action that occurs during the transition, and the posterior state. For a randomized system, the third element is replaced by a probability distribution over new states. An action π is said to be *enabled* after a finite execution α if there is a transition (s, π, P) such that s is the last state in α.

We will assume that for any state s and action π, there is at most one transition (s, π, P). We will call an action π a *deterministic* action if for any s such that (s, π, P) appears in the transition relation, P assigns probability 1 to a single state. Other actions are *randomized* actions. We will assume that a randomized action must be *local*: it can change the state of only one process.

Under the above assumptions, the effect of executing a deterministic action π after a finite execution α is well-defined; we will write the resulting execution as $\alpha\pi$. For a randomized action, suppose that C is a random variable representing its outcome; we will write αC for the (random) execution that results from executing the randomized action after α. In order to avoid dragging in too much measure-theoretic machinery, we will assume that there are only countably many possible outcomes of each local coin-flip. Among other things, this assumption means that we can exclude all outcomes whose probability is zero without having more than a probability-zero effect on the behavior of a system.

An execution β is an *extension* of α if α (considered formally as a sequence of states and actions) is a prefix of β. If the suffix of β after α consists only of deterministic actions, we write that β is a *deterministic extension* of α.

Often there will be several actions that are enabled after some finite execution α. The choice of which action to execute will be given to an *adversary*, a function mapping each finite execution to an action enabled in its final state. Letting the domain of the adversary function be the entire previous execution implies first that the adversary has total knowledge of the system's history and present state; but also that the adversary cannot base its choices on future events, such as the outcome of randomized actions that have not yet occurred. We will assume that the adversary does not itself use a randomized strategy; since we are in the lower-bound business this restriction on the adversary does

not affect our results.

3.2.2 Shared-Memory Model

The lower bound for randomized consensus will be given in the context of the standard asynchronous shared-memory model (see [Lyn96] for a definition of the shared-memory model in terms of I/O automata). In this model, the processes communicate by reading and writing a set of shared atomic registers. It may be assumed without loss of generality that operations on the registers are in fact instantaneous; so even though a read or write operation may be modeled formally as more than one action (e.g., as a separate invocation and response), we can treat this sequence of actions as a single *step*.

We will define the property that a step x is enabled after an execution α, the the result αx of executing x after α, and so forth, in the obvious way. Thus having secured the connection between our intuitive understanding and the underlying formal model, we will think of each process as carrying out a sequence of read, write, and local coin-flip steps, without worrying too much about the actual actions that make up these steps.

An additional property of the shared-memory model is that processes may fail. The failure of a process is a deterministic action that is always enabled, and its effect is to prevent the process from carrying out any more actions. We will usually assume a limit on the number of failures and require that any process that does not fail (or halt on its own) executes infinitely many steps. Both of these requirements are restrictions on the range of possible adversaries. The first forbids adversaries that cause too many failures. The second forbids adversaries that starve processes that have not failed.

An algorithm that operates in a model permitting up to t failures is called t-resilient.

3.3 Bivalence, Univalence, and Null-Valence

Formally, for each execution α and adversary A, write $\Pr[v|\alpha, A]$ for the probability that the protocol decides v after α running under the control of adversary A. For each execution α and set of adversaries $\mathcal{A}$, let $r_{v,\mathcal{A}}(\alpha)$ be the set of such probabilities ranging over all adversaries in $\mathcal{A}$; that is, $r_{v,\mathcal{A}} = \{\Pr[v|\alpha, A]|A \in \mathcal{A}\}$. Since the fact that the protocol terminates with probability 1 implies $\Pr[0|\alpha, A] = 1 - \Pr[1|\alpha, A]$, no additional information is gained by keeping track of r_0 and r_1 separately; thus we will drop the v subscript and write $r_{\mathcal{A}}(\alpha)$ for $r_{1,\mathcal{A}}(\alpha)$. In addition, when the set $\mathcal{A}$ is clear from context, we will drop it as well and just write $r(\alpha)$ for $r_{1,\mathcal{A}}(\alpha)$.

Fix $\epsilon > 0$. We will classify executions using the maximum probabilities of deciding 0 or 1 according to the following table.

An execution that is either 0-valent or 1-valent will be called univalent. Note that this classification is exhaustive: every execution falls into exactly one of these classes.

Classification of α	$\min r(\alpha)$	$\max r(\alpha)$
bivalent	$< \epsilon^2$	$> 1 - \epsilon^2$
0-valent	$< \epsilon^2$	$\leq 1 - \epsilon^2$
1-valent	$\geq \epsilon^2$	$> 1 - \epsilon^2$
null-valent	$\geq \epsilon^2$	$\leq 1 - \epsilon^2$

It is not hard to see that for deterministic algorithms these definitions reduce to the FLP definitions of a bivalent execution as one in which either outcome is possible (i.e., can occur with probability 1), and a v-valent execution is one in which only the outcome v is possible.

The FLP proof is based on the fact that for deterministic protocols, any extension of a v-valent execution is also v-valent. This fact is used to prove impossibility of deterministic consensus by showing that if a protocol can always extend a bivalent execution to either a 0-valent or 1-valent execution (necessary to reach a decision) it must have a 0-valent execution that is indistinguishable from a 1-valent execution (a contradiction).

We will not be deriving any contradictions from randomized protocols—randomized consensus is not impossible. Instead we will show that if a bivalent execution can be extended to either a 0-valent or a 1-valent execution through deterministic steps, then there exist deterministic extensions of these 0-valent and 1-valent executions that can be made indistinguishable. The resulting indistinguishable executions are not a contradiction; instead, they are null-valent.

The reason is that any deterministic extension of a v-valent execution may be either v-valent or null-valent. This fact is immediate from the lemma below:

Lemma 11 *Let α' be a deterministic extension of α. Then $r(\alpha') \subseteq r(\alpha)$.*

Proof: Let p be an element of $r(\alpha')$. Then there is some adversary A' in $\mathcal{A}$ such that $\Pr[1|\alpha', A'] = p$. But then the adversary A which first executes the steps leading to α' and then follows the strategy of A' gives $\Pr[1|\alpha, A] = p$, and thus p is in $r(\alpha)$. ■

If only deterministic operations are enabled after some execution α, the converse holds:

Lemma 12 *Let α be an execution after which only deterministic operations are enabled. Then $r(\alpha)$ is the union of $r(\alpha x)$ for each operation x enabled after α.*

Proof: In proving the lemma, it is necessary to be a little careful about failures. Observe that for any adversary A that fails some process after α, there is an adversary A' that simulates A without failing this process, delaying it instead until some other process decides. Since no process can distinguish A from A' until the decision value is fixed, both give the same probability of deciding 1 starting from α. Thus in computing $r(\alpha)$, we need consider only adversaries that do not fail any processes as the first step after α.

For each operation x enabled after α, let $\mathcal{A}_x$ be the set of adversaries that choose x. Then $r(\alpha) = \bigcup_x r_{\mathcal{A}_x}(\alpha) = \bigcup_x r(\alpha x)$. ■

In particular, if such an α only has v-valent successors, it must be v-valent.

In contrast, the range after a local coin-flip may be arbitrary. However, the *expected* endpoints of the range after the flip will always be equal to the endpoints of the range before the flip. This fact is not immediately obvious but it is not too hard to prove.

Lemma 13 *Let α be an execution and let C be a random variable that describes the outcome of some particular local coin-flip enabled after α. Then the expected value of $\min r(\alpha C)$ is equal to $\min r(\alpha)$, and the expected value of $\max r(\alpha C)$ is equal to $\max r(\alpha)$.*

Proof: We will prove the lemma only for $\max r(\alpha C)$; the case of $\min r(\alpha C)$ is symmetric.

First observe that $\max r(\alpha)$ is at least $\mathrm{E}[\max r(\alpha C)]$, since $\mathrm{E}[\max r(\alpha C)]$ is the probability of deciding 1 starting from α with an adversary that executes C and then follows the maximizing strategy in whatever execution results.

To show that $\max r(\alpha)$ is at most $\mathrm{E}[\max r(\alpha C)]$, let A be any adversary such that $\Pr[1|\alpha, A] = \max r(\alpha)$. We can modify A to get an adversary A' that executes C immediately after α, and then simulates A, ignoring the result of C until A chooses to execute C. Since a local coin-flip commutes with all operations of other processes, the executions produced by A and A' are indistinguishable and $\Pr[1|\alpha, A'] = \Pr[1|\alpha, A] = \max r(\alpha)$. But $\Pr[1|\alpha, A']$ is $\mathrm{E}[\Pr[1|\alpha C, A'] \leq \max r(\alpha C)$. ■

3.4 Valence of Initial States

To get the proof off the ground, we will need to show that there exists an initial state with the appropriate properties. The following lemma does so.

Lemma 14 *For any t-resilient consensus protocol with $t > 0$, there is an initial state α such that $\min r(\alpha) < \frac{1}{2}$ and $\max r(\alpha) \geq \frac{1}{2}$.*

Proof: The proof is essentially identical to the proof that a bivalent initial state exists for a deterministic protocol. First, observe that if two initial states α and α' differ in only one input, then given any adversary that kills the process with that input as the first action, the resulting executions are indistinguishable to all live processes. Thus $r(\alpha)$ and $r(\alpha')$ overlap at at least one point.

Now consider two states α and β such that $\min r(\alpha) = 0$ and $\max r(\beta) = 1$. (These states, which are not necessarily distinct, exist by the non-triviality condition.) There exists a chain of initial states $\alpha_0 = \alpha, \alpha_1, \alpha_2, \ldots, \alpha_k = \beta$ such that each adjacent pair of states differ in only one input. Let α_i be the first state in the chain for which $\max r(\alpha_i) \geq \frac{1}{2}$. If $i = 0$, we are done: $\min r(\alpha_i) = 0 < \frac{1}{2}$. Otherwise, $\max r(\alpha_{i-1}) < \frac{1}{2}$, implying $\min r(\alpha_i) < \frac{1}{2}$, since $r(\alpha_{i-1})$ and $r(\alpha_i)$ must overlap. ■

3.5 Strategy for Univalent Executions

Starting with a univalent execution, one of the outcomes can be forced with high probability. In this case the adversary's strategy will be to minimize the likelihood of that outcome in the hopes of getting back to a bivalent or null-valent execution. Its likelihood of being able to do so is described by the following lemma.

Lemma 15 *Let α be a failure-free 1-valent execution such that $\min r(\alpha) = p$. Then there is an adversary strategy that, with probability at least $1-p$, extends α to a failure-free execution $\alpha\beta$ such that one of the following conditions holds:*

1. *$\alpha\beta$ is null-valent;*
2. *$\alpha\beta$ is bivalent and β contains at least one local coin-flip;*
3. *$\alpha\beta$ is 0-valent, $\max r(\alpha\beta) > 1-\epsilon$, and β contains at least one local coin-flip; or*
4. *β contains an expected $1/\epsilon$ local coin-flips.*

Proof: Claim: For any failure-free execution α for which $0 < \min r(\alpha) < 1$, there exists some failure-free deterministic extension α' of α such that $\min r(\alpha') = \min r(\alpha)$ and a local coin-flip is enabled after α'. Proof: There is some adversary A for which $\Pr[1|\alpha, A] = \min r(\alpha)$. Since $\Pr[1|\alpha, A]$ is not 0 or 1, A must eventually cause the protocol to execute a coin-flip after some deterministic extension α' or α. (Otherwise the protocol does not satisfy the termination condition). If A causes failures, it can be simulated by an adversary A' that simply delays "failed" processes until after the coin-flip. That $\min r(\alpha') = \min r(\alpha)$ is immediate from Lemma 11.

Here is the full adversary strategy: carry out α' as described above, and then execute a local coin-flip. Repeat until $\max r$ drops to $1-\epsilon^2$, $\min r$ drops below ϵ^2, or a decision is reached.

Now let us show that this strategy works as advertised. We have $\min r(\alpha') = \min r(\alpha) \geq 1-\epsilon^2$; and from Lemma 11 the only other effect of executing deterministic steps can be to reduce $\max r(\alpha')$. If $\max r(\alpha')$ is less than or equal to $1-\epsilon^2$, α' is null-valent and case (1) of the lemma holds. Otherwise, we must consider the possible outcomes of the local coin-flip.

Let C be the random variable whose values are the possible outcomes of the local coin-flip. From Lemma 13, $\mathrm{E}[\max r(\alpha' C)] = \max r(\alpha) > 1-\epsilon^2$. By Markov's inequality, the probability that $\max r(\alpha' C)$ is less than or equal to $1-\epsilon$ is less than ϵ. So on average, we expect to execute at least $1/\epsilon$ local coin-flips before $\max r$ drops below $1-\epsilon$. This possibility accounts for case (4).

Suppose that $\max r(\alpha C)$ does not drop below $1-\epsilon$. There are several possibilities. If $\min r(\alpha C) < \epsilon^2$, case (2) or (3) holds, and we are done. If $\min r(\alpha C) \geq \epsilon^2$ and $\max r(\alpha C) \leq 1-\epsilon^2$, case (1) holds, and again we are done. If $\max r(\alpha C) > 1-\epsilon^2$, then we may repeat the process described above until we reach a new local coin-flip, unless a decision of 1 is reached; but the probability

that the protocol reaches a decision of 1 following this strategy is at most p. So with probability at least $1 - p$, one of the other outcomes occurs. ■

By symmetry, it is immediate that the lemma also holds if the decision values 0 and 1 are swapped.

3.6 Strategy for Bivalent Executions

Given a bivalent execution α, we wish to show that α has a fair, failure-free deterministic extension that is either null-valent (so we can apply the coin-flipping bound) or permits a local coin-flip in its final state.

Lemma 16 *Let α be a failure-free bivalent execution, and let x be a deterministic operation that is enabled after α. Then there exists a finite deterministic extension β of α such that one of the following conditions holds:*

1. *β is failure-free, bivalent, and a local coin-flip is enabled after β;*
2. *β is failure-free, bivalent, and contains x; or*
3. *β contains at most one failure and is null-valent.*

Proof: Consider the set S of all bivalent failure-free deterministic extensions γ of α. If there is a γ in S after which a local coin-flip is enabled, we are done: case (a) holds with $\beta = \gamma$. If there is a γ in S containing x, we are done: case (2) holds with $\beta = \gamma$. If there is a γ in S such that γy is null-valent for some y, we are done: case (3) holds with $\beta = \gamma y$.

Otherwise, let γ be a maximal execution in S, i.e. one such that no extension of γ is in S. Such an execution exists because every execution in S must be finite by the termination condition. Under the assumption that none of the conditions above hold, we know that:

1. No local coin-flip is enabled after γ.
2. For each operation y, γy is univalent. (It cannot be null-valent; nor can it be bivalent, because then γ is not maximal.)
3. The operation x is enabled after γ. (It is enabled after α; for it not to be enabled after γ it must appear in γ.)

Assume without loss of generality that γx is 0-valent; the case where it is 1-valent is symmetric. Then there exists some y such that γy is 1-valent, as otherwise γ would be 0-valent by Lemma 12, contradicting its membership in S. We will show by a case analysis that there are always deterministic extensions of γx and γy that are distinguishable by at at most one process. Killing this process thus leads to indistinguishable executions that deterministically extend 0-valent and 1-valent executions; these executions must both be null-valent and we can choose either one for β and satisfy condition (3).

There are several cases depending on the type of the operations x and y:

1. x and y are operations on different registers. In this case x and y commute and the states resulting from γxy and γyx are the same. No killing is necessary; γxy and γyx are both null-valent.

2. x is a read operation. If y is also a read operation, the operations commute and γxy and γyx are both null-valent as above. If y is a write operation, then only the process performing x can distinguish between γyx and γxy. Killing this process yields a pair of indistinguishable executions that are thus both null-valent.

3. y is a read operation. This case is symmetric with the previous case.

4. x and y are both write operations on the same register. Then γyx is distinguishable from γx only by the process performing y. Again, killing this process yields two indistinguishable executions that must both be null-valent.

■

Iterating the lemma eliminates one of the cases:

Lemma 17 *Let α be a failure-free bivalent execution. Then there exists a finite deterministic extension β of α such that either*

1. *β is failure-free and a local coin-flip is enabled after β, or*

2. *β contains at most one failure and is null-valent.*

Proof: Let $\alpha_0 = \alpha$. For each α_i, let x_i be the operation enabled after α_i that has been enabled the longest and let α_{i+1} be the finite deterministic extension of α_i whose existence is implied by Lemma 16. If α_{i+1} is failure-free and a local coin-flip is enabled after it, set $\beta = \alpha_{i+1}$. Similarly set $\beta = \alpha_{i+1}$ if α_{i+1} contains at most one failure and is null-valent. If α_{i+1} is bivalent, contains x_i, and no coin-flip is enabled in α_{i+1}, continue with α_{i+1} and a new x_{i+1}.

This process must eventually terminate with β equal to some α_i. Otherwise, it would yield an infinite, fair, failure-free deterministic extension of α, violating the termination condition. ■

The lemma above implies that we can always reach an execution that either is null-valent or permits a coin-flip. For this fact to be useful, coin-flips cannot be too destructive. The following lemma constrains their wrath:

Lemma 18 *Let α be a bivalent execution and let C be a random variable corresponding to the possible outcomes of a local coin-flip enabled in α. Then*

$$\Pr\left[\min r(\alpha C) \geq \epsilon \vee \max r(\alpha C) \leq 1 - \epsilon\right] < 2\epsilon.$$

Proof: Let $X = \min r(\alpha C)$. Then $X \geq 0$, and $\mathrm{E}[X] = \min r(\alpha) < \epsilon^2$. So by Markov's inequality the probability that $\min r(\alpha C)$ reaches ϵ is less than $\epsilon^2/\epsilon = \epsilon$. Adding the probability of the symmetric event that $\max(r\alpha C)$ reaches $1 - \epsilon$ raises the bound to 2ϵ. ■

Thus with high probability, the result of a local coin-flip after a bivalent execution is an execution that is either bivalent, null-valent, or univalent with a very wide range. In the case of a bivalent execution the adversary may continue as above. In the case of a null-valent execution, the coin-flipping bound applies. The case of a univalent execution with a wide range is covered below.

3.7 The Full Strategy

Here is the full strategy used to prove the lower bound. It is divided into four cases corresponding to four conditions that could hold at the end of each partial execution:

1. Start in an initial state whose range straddles $\frac{1}{2}$. The existence of such a state is guaranteed by Lemma 14. If this state is bivalent or null-valent, skip to the appropriate condition below. Otherwise, the state must either be 0-valent with $\max r \geq \frac{1}{2}$ or 1-valent with $\min r \leq \frac{1}{2}$. In either case, Lemma 15 or its symmetric equivalent applies, and with probability at least $\frac{1}{2}$ we reach one of conditions (2), (3), or (4); or we execute an expected $1/\epsilon$ local coin-flips before deciding.

2. From a 0-valent execution with $\max r > 1 - \epsilon$ or a 1-valent execution with $\min r < 1 - \epsilon$, apply Lemma 15 or its symmetric equivalent. With probability $1 - \epsilon$: one of the following occurs: we reach one of conditions (2) or (3) after executing at least one local coin-flip; we reach condition (4); or we execute an expected $1/\epsilon$ local coin-flips before deciding.

3. From a bivalent execution, apply Lemma 17 to either reach a null-valent execution with at most one failure, or a bivalent execution after which a local coin-flip is enabled. If we reach a bivalent execution after which a local coin-flip is enabled, apply Lemma 18 to show that the result of this coin-flip lands us in condition (2), (3), or (4) with probability at least $1 - 2\epsilon$.

4. From a null-valent execution with at most one failure, Theorem 10 applies, and there is an adversary strategy that forces an expected $\frac{3(t-1)^2}{64 \ln^2\left(\frac{2}{\epsilon}-1\right)}$ local coin-flips before termination.

With a suitable choice of ϵ, the existence of this strategy implies:

Theorem 19 *Against a worst-case adaptive adversary, any t-resilient consensus protocol for the asynchronous shared-memory model performs an expected*

$$\Omega\left(\left(\frac{t-1}{\log(t-1)}\right)^2\right)$$

local coin-flips.

Proof: Let I be the expected cost from the initial state in case (1); U the expected cost from the univalent execution in case (2); B the expected cost from the bivalent execution in case (3); and N the expected cost from the null-valent execution in case (4). Then we have:

$$\begin{aligned} I &\geq \frac{1}{2}\min(U, B, N, 1/\epsilon) \\ U &\geq (1-\epsilon)\min(1+U, 1+B, N, 1/\epsilon) \\ B &\geq \min(N, (1-2\epsilon)(1+\min(U, B, N))) \\ N &\geq \frac{3(t-1)^2}{64\ln^2\left(\frac{2}{\epsilon}-1\right)}. \end{aligned}$$

Our goal is to work backwards from the lower bound on N to get a lower bound for I. Let M be the smallest of U, B, and N. There are three cases:

- $M = U$. Then $U \geq (1-\epsilon)\min(1+U, 1/\epsilon)$. This implies U is at least $1/\epsilon - 1$; for if we assume that $U \leq 1/\epsilon - 1$ we get $U \geq (1-\epsilon)(1+U)$ and thus $\epsilon U \geq 1-\epsilon$ or $U \geq 1/\epsilon - 1$.
- $M = B$. Then $B \geq (1-2\epsilon)(1+B)$, implying $B \geq \frac{1}{2\epsilon} - 1$.
- $M = N$. Then $M \geq \frac{3(t-1)^2}{64\ln^2\left(\frac{2}{\epsilon}-1\right)}$.

In each case we have

$$M \geq \min\left[\frac{1}{2\epsilon} - 1, \frac{3(t-1)^2}{64\ln^2\left(\frac{2}{\epsilon}-1\right)}\right] \tag{14}$$

Since $I \geq \frac{1}{2}\min(M, 1/\epsilon)$, the right-hand side of (14), divided by 2, gives a lower bound on the number of local coin-flips executed by the consensus protocol. If we set $\epsilon = (t-1)^{-2}$, this expression reduces to the bound claimed in the theorem. ■

The bound counts the number of local coin-flips. Because we allow coin-flips to have arbitrary values (not just 0 or 1), local coin-flips performed by the same process without any intervening operations can be combined into a single coin-flip without increasing the adversary's influence. Thus the lower bound on local coin-flips immediately gives a lower bound on total work. Furthermore, because the coin-flip bound is not affected by changing the model to one that can be deterministically simulated by shared memory, we get the same lower bound on total work in any model that can be so simulated, no matter how powerful its primitives are. So, for example, wait-free consensus requires $\Omega(n^2/\log^2 n)$ work even in a model that supplies counters or $O(1)$-cost atomic snapshots.

4 Discussion

For those of us who like working with an adaptive adversary, randomization has given only a temporary reprieve from the consequences of Fischer, Lynch, and

Paterson's impossibility proof for deterministic consensus with faulty processes. Theorem 19 means that even though we can solve consensus using randomization, we cannot hope to solve it quickly without a small upper bound on the number of failures, built-in synchronization primitives, or restrictions on the power of the adversary.

Fortunately, there are a number of natural restrictions on the adversary that allow fast consensus protocols without eliminating the faults that we might reasonably expect to observe in real systems. One plausible approach is to limit the knowledge the adversary has of register contents, to prevent it from discriminating against coin-flips it dislikes. Various versions of this can be found in the the consensus work of Chor, Israeli, and Li [CIL87] and Abrahamson [Abr88], and in the $O(n \log^2 n)$ total work protocol of Aumann and Bender [AB96], the $O(\log^2 n)$ work-per-process protocol of Chandra [Cha96], and the recent $O(\log n)$ work-per-process protocol of Aumann [Aum97]. Restrictions on the amount of asynchrony can also have a large effect [AAT94, SSW91].

A question that we have not completely answered is the following: Does the majority of n fair coin-flips give an optimal coin-flipping game (in the sense of having minimum bias) with an adversary that can censor up to k flips? Majority is optimal for similar models (e.g., in the fair-local-coin model studied by Lichtenstein, Linial, and Saks [LLS89]). Theorem 3 implies that it is not possible to achieve constant bias with more than $k = O(\sqrt{n})$ faults, the amount tolerated by majority, but there is still a gap between the lower bound of Theorem 3 and the upper bound of the majority game when k is large relative to $\sqrt{n}$. It can be shown using the analysis in Section 2.1 that taking a majority of fair coins cannot be optimal in an absolute sense when biased local coins are allowed, as the optimal games characterized in that section generally do not use fair local coins. However it is still possible that majority is close to optimal, and it might be possible to show (for example) that no game where the adversary was allowed to hide $2k$ local coins could have a smaller bias than majority with k hidden coins.

One possible approach to showing majority is close to optimal is suggested by the fact that the hyperbolic tangent function tanh in Theorem 3 essentially acts as an easier-to-manipulate approximation to the normal distribution function Φ. If we replace tanh by Φ and adjust the set of possible game outcomes to match the range of Φ, we get the following conjecture:

Conjecture 20 *Let G be a game of length n with outcome set $\{0, 1\}$. Then there exists a constant $c > 0$ such that for any $k \geq 0$, either $M_k G = 1$, $m_k G = 0$, or*

$$\Phi^{-1} M_k G - \Phi^{-1} m_k G \geq \frac{ck}{\sqrt{n}}. \tag{15}$$

If true, the conjecture would give a lower bound that would match (up to constant factors) the upper bound given by majority voting, and would improve by a factor of $\log n$ the lower bound for consensus given in Theorem 19.

A still more general question asked by Ben-Or and Linial in [BOL89], also still open, is whether majority voting is optimal in a Byzantine model where

processes may vote more than once but in which the adversary controls all future votes of a process once it has been corrupted. Our work shows that the number of local coins flipped in this model must be large relative to the number of failures, but it does not exclude the possibility that the number of distinct processes might still be relatively small.

5 Acknowledgments

The author is indebted to Russell Impagliazzo for many fruitful discussions of coin-flipping problems, Steven Rudich for a suggestion that eventually became the truncation argument used to prove Theorem 10, Mike Saks for encouragement and pointers to related work, and Faith Fich, Wai-Kau Lo, Eric Ruppert, and Eric Schenk for many useful comments on an earlier version of this work.

References

[AAT94] Rajeev Alur, Hagit Attiya, and Gadi Taubenfeld. Time-adaptive algorithms for synchronization. In *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing*, pages 800–809, Montréal, Québec, Canada, may 1994.

[AB96] Yonatan Aumann and Michael Bender. Efficient asynchronous consensus with a value-oblivious adversary scheduler. In *Proceedings of the 23rd International Conference on Automata, Languages, and Programming*, July 1996.

[Abr88] K. Abrahamson. On achieving consensus using a shared memory. In *Proceedings of the Seventh ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing*, August 1988.

[ADS89] Hagit Attiya, Danny Dolev, and Nir Shavit. Bounded polynomial randomized consensus. In *Proceedings of the Eighth ACM Symposium on Principles of Distributed Computing*, pages 281–294, August 1989.

[AH90] James Aspnes and Maurice Herlihy. Fast randomized consensus using shared memory. *Journal of Algorithms*, 11(3):441–461, September 1990.

[AN90] Noga Alon and Moni Naor. Coin-flipping games immune against linear-sized coalitions. In *Proceedings of the 31st Annual Symposium on Foundations of Computer Science*, pages 46–54. IEEE, 1990.

[Asp93] James Aspnes. Time- and space-efficient randomized consensus. *Journal of Algorithms*, 14(3):414–431, May 1993.

[Asp97] James Aspnes. Lower bounds for distributed coin-flipping and randomized consensus. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing*, pages 559–568. ACM, May 1997.

[Aum97] Yonatan Aumann. Efficient asynchronous consensus with the weak adversary scheduler. In *Proceedings of the Sixteenth Annual ACM Symposium on Principles of Distributed Computing*, pages 209–218, 1997.

[AW96] James Aspnes and Orli Waarts. Randomized consensus in $O(n \log^2 n)$ operations per processor. *SIAM Journal on Computing*, 25(5):1024–1044, October 1996.

[BOL89] Michael Ben-Or and Nathan Linial. Collective coin flipping. In Silvio Micali, editor, *Randomness and Computation*, volume 5 of *Advances in Computing Research*, pages 91–115. JAI Press, 1989.

[BOLS87] M. Ben-Or, N. Linial, and M. Saks. Collective coin flipping and other models of imperfect randomness. In *Combinatorics*, volume 52 of *Colloquia Mathematic Societatis János Bolyai*, pages 75–112, Eger (Hungary), 1987.

[BR90] Gabi Bracha and Ophir Rachman. Approximated counters and randomized consensus. Technical Report 662, Technion, 1990.

[BR91] Gabi Bracha and Ophir Rachman. Randomized consensus in expected $O(n^2 \log n)$ operations. In *Proceedings of the Fifth Workshop on Distributed Algorithms*, 1991.

[CFG+85] Benny Chor, Joel Friedman, Oded Goldreich, Johan Håstad, Steven Rudich, and Roman Smolensky. The bit extraction problem or t-resilient functions. In *Proceedings of the 2th Annual Symposium on Foundations of Computer Science*, pages 396–407. IEEE, 1985.

[Cha96] Tushar Deepak Chandra. Polylog randomized wait-free consensus. In *Proceedings of the Fifteenth Annual ACM Symposium on Principles of Distributed Computing*, pages 166–175, May 1996.

[CI93] Richard Cleve and Russell Impagliazzo. Martingales with Boolean final value must make jumps of $O(1/n^{1/2})$ with constant probability. Unpublished manuscript, 1993.

[CIL87] B. Chor, A. Israeli, and M. Li. On processor coordination using asynchronous hardware. In *Proceedings of the Sixth ACM Symposium on Principles of Distributed Computing*, pages 86–97, 1987.

[CL93] Jason Cooper and Nathan Linial. Fast perfect-information leader-election protocol with linear immunity. In *Proceedings of the Twenty-Fifth Annual ACM Symposium on the Theory of Computing*, pages 662–671. ACM, 1993.

[DDS87] D. Dolev, C. Dwork, and L. Stockmeyer. On the minimal synchronism needed for distributed consensus. *Journal of the ACM*, 34(1):77–97, January 1987.

[DHPW92] Cynthia Dwork, Maurice Herlihy, Serge Plotkin, and Orli Waarts. Time-lapse snapshots. In *Proceedings of Israel Symposium on the Theory of Computing and Systems*, 1992.

[FHS93] Faith Fich, Maurice Herlihy, and Nir Shavit. On the complexity of randomized synchronization. In *Proceedings of the 12th Annual ACM Symposium on Principles of Distributed Computing*, August 1993.

[FLP85] M. Fischer, N.A. Lynch, and M.S. Paterson. Impossibility of distributed commit with one faulty process. *Journal of the ACM*, 32(2), April 1985.

[Fri92] Joel Friedman. On the bit extraction problem. In *Proceedings of the 33rd Annual Symposium on Foundations of Computer Science*, pages 314–319. IEEE, 1992.

[Har66] L. H. Harper. Optimal numberings and isoperimetric problems on graphs. *Journal of Combinatorial Theory*, 1:385–394, 1966.

[Her91] Maurice Herlihy. Wait-free synchronization. *ACM Transactions on Programming Languages and Systems*, 13(1):124–149, January 1991.

[LAA87] Michael C. Loui and Hosame H. Abu-Amara. Memory requirements for agreement among unreliable asynchronous processes. In Franco P. Preparata, editor, *Advances in Computing Research*, volume 4. JAI Press, 1987.

[LLS89] D. Lichtenstein, N. Linial, and M. Saks. Some extremal problems arising from discrete control processes. *Combinatorica*, 9:269–287, 1989.

[Lyn96] Nancy A. Lynch. *Distributed Algorithms*. Morgan Kaufmann, 1996.

[Sak89] Michael Saks. A robust non-cryptographic protocol for collective coin flipping. *SIAM Journal on Discrete Mathematics*, 2(2):240–244, 1989.

[SSW91] Michael Saks, Nir Shavit, and Heather Woll. Optimal time randomized consensus — making resilient algorithms fast in practice. In *Proceedings of the Second Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 351–362, 1991.

[Vaz85] Umesh Vazirani. Towards a strong communication complexity theory, or generating quasi-random sequences from two communicating slightly-random sources. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing*, pages 366–378. ACM, 1985.