Average Binary Long-Lived Consensus and the Stabilizing Role Played by Memory *

Florent Becker[1], Sergio Rajsbaum[2], Ivan Rapaport[3] and Eric Rémila[1]

[1]Laboratoire de l'Informatique du Parallélisme, École Normale Supérieure de Lyon, France.
[2]Instituto de Matemáticas, Universidad Nacional Autónoma de México.
[3]Departamento de Ingeniería Matemática and Centro de Modelamiento Matemático, Universidad de Chile.

Abstract

Consider a system composed of n sensors operating in synchronous rounds. In each round an *input vector* of sensor readings x is produced, where the i-th entry of x is a binary value produced by the i-th sensor. The sequence of input vectors is assumed to be *smooth*: exactly one entry of the vector changes from one round to the next one. The system implements a fault-tolerant averaging *consensus function* f. This function returns, in each round, a representative *output value* v of the sensor readings x. Assuming that at most t entries of the vector can be erroneous, f is required to return a value that appears at least $t+1$ times in x. The *instability* of the system is the number of output changes over a random sequence of input vectors.

Our first result is to design optimal instability consensus systems with and without memory. Intuitively, in the memoryless case, we show that an optimal system is D_0, that outputs 1 unless it is forced by the fault-tolerance requirement to output 0 (on vectors with t or less 1's). For the case of systems with memory, we show that an optimal system is D_1, that initially outputs the most common value in the input vector, and then stays with its output unless forced by the fault-tolerance requirement to change (i.e., a single bit of memory suffices).

Our second result is to quantify the gain factor due to memory by computing $c_n(t)$, the number of decision changes performed by D_0 per each decision change performed by D_1. If $t = \frac{n}{2}$ the system is always forced to decide the simple majority and, in that case, memory becomes useless. We show that the same type of phenomenon occurs when $\frac{n}{2} - t$ is constant. Nevertheless, as soon as $\frac{n}{2} - t \sim \sqrt{n}$, memory plays an important stabilizing role because the ratio $c_n(t)$ grows like $\Theta(\sqrt{n})$. We also show that this is an upper bound: $c_n(t) = O(\sqrt{n})$.

Our results are average case versions of previous works where the sequence of input vectors was assumed to be, in addition to smooth, *geodesic*: the i-th entry of the input vector was allowed to change *at most once* over the sequence. It thus eliminates some anomalies that ocurred in the worst case, geodesic instability setting.

1 Introduction

Consider a system composed of n sensors sampled at synchronous rounds. In each round an *input vector* of sensor readings is produced, where the i-th entry of the vector is a value from some finite

*Partially supported by Programs Conicyt "Anillo en Redes", Instituto Milenio de Dinámica Celular y Biotecnología and Ecos-Conicyt.

set V produced by the i-th sensor. To simplify the presentation, the sampling interval is assumed to be short enough, to guarantee that the sequence of input vectors is *smooth*: exactly one entry of a vector changes from one round to the next one.

There are situations where, for fault-tolerant purposes, all sensors are placed in the same location. Ideally, in such cases, all sensor readings should be equal. But this is not always the case; discrepancies may arise due to differences in sensor readings or to malfunction of some sensors. Thus, the system must implement some form of fault-tolerant averaging *consensus function* f, that returns a representative *output value* v of the sensor readings x. Assuming that at most t entries of a vector x can be erroneous, f is required to return a value that appears at least $t+1$ times in x.

A natural function f is the one that returns the most common value of vector x. However, the *instability* of such function is high. In fact, as the next example shows ($n = 5$ and $t = 1$), the output value computed by this f could change from one round to the next one unnecessarily often:

inputs:	00011	$\rightarrow$	10011	$\rightarrow$	10010	$\rightarrow$	11010	$\rightarrow \cdots$
outputs:	0	$\rightarrow$	1	$\rightarrow$	0	$\rightarrow$	1	$\rightarrow \cdots$

If instead of the previous f we consider the one that decides the smallest value in x that appears at least $t+1$ times, then no output changes would have occurred in the previous sequence (in the example $0 < 1$ and $t + 1 = 2$). Moreover, we could consider a function f *with memory* to reduce further the instability, and try to stay with the ouput of previous rounds.

The worst case instability of consensus functions was studied in two previous papers [3, 5]. The input sequence considered in those papers was assumed to be, in addition to smooth, *geodesic*: the i-th entry of the input vector was allowed to change *at most once* over the sequence. The instability of a consensus function was given by the largest number of output changes over any such sequence, called a *geodesic path*. Notice that a geodesic path must be finite, since the set V from which the input vectors draw their values is finite. The case $V = \{0, 1\}$ of binary input vectors was considered in [5]. The case of multi-valued input vectors, where the set V is arbitrary, turned out to be much more difficult and required higher-dimensional topological methods [3].

In this paper we initiate a study of the *average instability* of consensus functions. We tackle the case $V = \{0, 1\}$ of binary input vectors. We remove the geodesic requirement and therefore the smooth sequences of input vectors we consider here are random walks over the hypercube. If $P = X_0, X_1, \ldots$ is such a walk, then the average instability of a consensus function f is given by the fraction of time f changes its output over P. The first goal is –given n and t– to find out what function f minimizes the instability in the two possible scenarios: with memory and without memory. We obtain the following results.

For the memoryless case we show that a system, D_0, that outputs 1 unless it is forced by the fault-tolerance requirement to output 0 (on vectors with t or less 1's), is optimal. For the case of systems with memory, we show that a system, D_1, that initially outputs the most common value in the input vector, and then stays with its output unless forced by the fault-tolerance requirement to change is optimal. Thus, a single bit of memory suffices to achieve optimal instability. We should point out that in order to compute the instability of D_1 we use a non-trivial result concerning the Ehrenfest Markov chain model (which gives a simple expression to the value of the expected time to go from state k to state $k+1$ [14]).

Our second goal is to measure the stabilizing role played by memory. A natural way of doing this is by computing $c_n(t)$, the number of decision changes performed by D_0 per each decision change performed by D_1. We prove that $c_n(t) = O(\sqrt{n})$, and this upper bound is reached when

$n/2-t = \alpha\sqrt{n}$, with α constant. In contrast, if t or $n/2-t$ are constant then $c_n(t)$ is also constant.

Our approach eliminates some anomalies that occured in the worst case geodesic instability setting. For instance, in the case of $t = 0$ (which is interesting because it leaves maximum freedom on the choice of f), it was proved in [5] that any optimal instability memoryless function must be *one-bit defined*, i.e., output the value of the i-th sensor. Intuitively, such a function has high instability. Indeed, in our average case setting, its instability is $1/n$, much higher than the optimal average instability of $1/2^n$ given by f_0.

As noted in [5], studying the instability of consensus functions may have applications in various areas of distributed computing, such as self-stabilization [4] (indeed, see [10]), Byzantine agreement [1], real-time systems [11], complexity theory [8] (boolean functions), and VLSI energy saving [2, 12, 16] (minimizing number of transitions).

Also, it would be interesting to relate our results to natural phenomena that exhibit hysteresis (memory). It is known, for instance, that some biology phenomena exhibit hysteresis [9, 15]. Did they appear in evolution as a way to minimize instability? The approach could also be applied in the social sciences. In fact, wouldn't it be possible to conceive an electoral system which instead of deciding by simple majority incorporates some memory in order to eliminate noise? Notice that system D_1, which minimizes instability, corresponds to an hysteretic switch.

This paper is organized as follows. In Section 2 we describe the model and define the instability measure. Section 3 considers memoryless consensus systems, while Section 4 considers the general case. In Section 5 we quantify the relevance of memory by analyzing the behavior of the gain factor for different values of t.

2 Instability

Let n, t be non-negative integers, $n \geq 2t+1$. The *hypercube* of dimension n is a graph whose vertices $V_n = \{0,1\}^n$ are all binary n-dimensional vectors, called *input vectors*. The edges E_n are all pairs of vertices whose vectors differ in exactly one component. Notice that $|E_n| = n2^{n-1}$. The *distance* $d(x_1, x_2)$ between two vertices x_1, x_2 is equal to the number of entries in which they differ. Thus, $d(x_1, x_2) = d$ iff the shortest path between x_1 and x_2 in the hypercube is of length d. We denote by $\#_b(x)$ the number of entries in x that are equal to $b \in \{0,1\}$. The *corners* of the hypercube are the vertices 0^n and 1^n. The *d-neighborhood* of a vertex x of the hypercube, $\mathcal{N}^d(x)$, is the set of vertices at distance at most d from x. Thus, $\mathcal{N}^t(0^n) = \{x \mid \#_1(x) \leq t\}$, and similarly for 1^n. Since $n \geq 2t+1$, $\mathcal{N}^t(0^n) \cap \mathcal{N}^t(1^n) = \emptyset$.

Let $x_0, x_1, \ldots$ be the vertices of a walk in the hypercube. We will consider functions f that assign, to each x_i, an *output value* d that satisfies the *fault-tolerance* requirement:

$$f(x_i) = d \Rightarrow \#_d(x_i) \geq t+1.$$

In the *memoryless* case f is a function only of x_i. In general, f outputs d based on the previous vertices of the walk. Formally, a *system* is a tuple $D = (n, t, S, \tau, f)$, where S is a finite set of states that includes a special initial state $\perp \in S$, $\tau : V_n \times S \to S$ is the transition function, and $f : V_n \times S \to \{0,1\}$ is the consensus decision function. The fault-tolerance requirement implies, for all $x \in V_n$, $s \in S$:

$$f(x,s) = \begin{cases} 0 & \text{if } x \in \mathcal{N}^t(0^n) \\ 1 & \text{if } x \in \mathcal{N}^t(1^n) \end{cases}$$

An *execution* of the system is a sequence $(x_0, s_0, d_0) \to (x_1, s_1, d_1) \to \ldots$, where $s_0 = \perp$, $s_{i+1} = \tau(x_i, s_i)$, and $d_i = f(x_i, s_i)$. A triple (x_i, s_i, d_i) is a *configuration*.

We assume that if x is the current input vector, then the next input vector x' is taken in a random uniform way from the vectors at distance one from x in the hypercube. The initial input vector is chosen according to some distribution λ. Once the initial state x_0 is determined, so is the initial configuration, (x_0, s_0, d_0). The next configuration is produced by choosing at random a neighbor of x_0, say x_1, and we get the next configuration (x_1, s_1, d_1), where $s_1 = \tau(x_0, s_0)$ and $d_1 = f(x_1, s_1)$.

Formally, we define the following Markov process: the set of states is $V_n \times S$ and there is a transition from (x, s) to (x', s') if $\{x, x'\} \in E_n$ and $\tau(x, s) = s'$. Therefore, any random walk $X_0, X_1, X_2, \ldots$, where X_0 is chosen according to λ, defines an execution.

Each state (x, s) has an associated output value $f(x, s)$, so we may write $d_i = f(X_i)$ to be the output value associated to X_i. Let $c_{\lambda,l}(D)$ be the random variable defined by:

$$c_{\lambda,l}(D) = \frac{1}{l} \sum_{k=0}^{l-1} |d_{k+1} - d_k|.$$

Definition 1. *The* average instability *of a consensus system D is* $c(D) = \mathbb{E}(\lim_{l\to\infty} c_{\lambda,l}(D))$.

The Markov chain described above is finite and hence $c(D)$ exists by the ergodic theorem[1].

3 Average instability of memoryless systems

In a memoryless system $|S| = 1$, and τ is irrelevant, so the system is defined by a triplet $D = (n, t, f)$, where $f : V_n \to \{0, 1\}$. In this case, the Markov chain is irreducible[2], and its set of states is V_n. That is, $X_0, X_1, X_2, \ldots$ is a λ-random walk on the hypercube, and its stationary distribution π is the uniform $\pi_x = 1/2^n$, for every $x \in V_n$ (for notation see [13])). The instability of f counts the number of times the function f changes its decision along a random walk. By the ergodic theorem, the fraction of time the random walk crosses bicolored edges (where changes in the decision take place) tends to the number of bicolored edges divided by $|E_n| = n2^{n-1}$.

Proposition 1. *Let $D = (n, t, f)$ be a memoryless system. Then,*

$$c(D) = \frac{\sum_{\{x,y\}\in E_n} |f(x) - f(y)|}{n2^{n-1}}$$

Proof. Since by definition $c(D) = \mathbb{E}(\lim_{l\to\infty} c_{\lambda,l}(D))$, it is sufficient to prove that

$$c_{\lambda,l}(D) \underset{l\to\infty}{\longrightarrow} \frac{\sum_{\{x,y\}\in E_n} |f(x) - f(y)|}{n2^{n-1}} \quad \text{a.s.}$$

This follows directly from the ergodic theorem. Formally, we consider the Markov chain (X_0, X_1), $(X_1, X_2), \ldots$ in which each state is an arc $e = (X_i, X_{i+1}) \in \overrightarrow{E}_n$, an ordered couple of neighbor vertices of the hypercube. Therefore the function defined over the set of states (the arcs) is

[1] When S is not finite, $c(D)$ might not exist, but $c'(D) = \mathbb{E}(\liminf_{l\to\infty} c_{\lambda,l}(D))$ always exists, and can be considered as a lower bound for the cost.

[2] In an irreducible Markov chain, it is possible to get to any state from any state.

$f(X_i, X_{i+1}) = |f(X_{k+1}) - f(X_k)|$. In this standard approach one needs to verify that the chain is irreducible and that the unique invariant distribution is the uniform distribution in order to conclude that:

$$c_{\lambda,l}(D) = \frac{1}{l}\sum_{k=0}^{l-1} f(X_k, X_{k+1}) \underset{l\to\infty}{\longrightarrow} \sum_{e\in\overrightarrow{E}_n} \pi_e f(e) = \frac{1}{2n2^{n-1}} \sum_{e\in\overrightarrow{E}_n} f(e) \quad = \frac{1}{n2^{n-1}} \sum_{e\in E_n} f(e) \qquad \text{a.s.}$$

□

3.1 Geodesic worst case vs. average instability

The worst case geodesic instability measure of [5, 3] depends on the values f takes in a small part of the hypercube, given by a geodesic path (where the i-th entry of a vector changes at most once). In contrast, the average instability permits walks that traverse the whole hypercube. We are going to remark this difference by giving two examples for which the values behave in opposite ways.

Let us assume $t = 0$. In this case the only restrictions appear in the corners of the hypercube. More precisely, $f(d^n) = d$.

Suppose that the output of f depends exclusively on what happens in one particular processor (the i-th processor). In other words, consider the function $f^{(i)}(x) = x^{(i)}$, where $x = x^{(1)} \dots x^{(n)}$. By common sense, this is clearly a bad strategy. But in terms of the geodesic analysis, this function appeared to be optimal [5]. Morover, it was proved that any optimal function must be of this form (when $t = 0$). The explanation comes from the fact that in a geodesic path, once the i-th coordinate changes, it can not change anymore. On the other hand, by Proposition 1, the average instability of $f^{(i)}$ is $c(f^{(i)}) = \frac{1}{n2^{n-1}}\frac{2^n}{2} = \frac{1}{n}$.

It is easy to see that the average instability of $f^{(i)}$ is far from being optimal. For example, assume n is odd. And let $f(x) = 1$ if and only if $x = 0^k1^l$ with l odd. There is a geodesic path for which the function changes *all along the path*. Nevertheless, there is a small number of 1's in the hypercube. And, in fact, $c(f) = \frac{1}{n2^{n-1}}\frac{n}{2}n = \frac{n}{2^n} << c(f^{(i)})$.

3.2 Optimal memoryless systems

Let Γ_i denote the set of vectors $x \in V_n$ satisfying $\#_1(x) = i$. In other words, Γ_i is the set of nodes of the hypercube at distance i from 0^n. Recall that $\mathcal{N}^k(x) = \{y \mid d(x, y) \le k\}$. Let us define $D_0 = (n, t, f_0)$ with $f_0(x) = 0$ if and only if $x \in \mathcal{N}^t(0^n)$. In other words, f_0 is always 1 unless the fault-tolerance requirement force the system to decide 0. We prove below that D_0 is an optimal memoryless system.

Theorem 1. *Let $D = (n, t, f)$ be a memoryless system. Then $c(D_0) \le c(D)$, with $c(D_0) = \frac{\binom{n-1}{t}}{2^{n-1}}$.*

Proof. By Proposition 1, in order to compute $c(D_0)$, we need to count the number of bicolored edges $\{x, y\}$ induced by f_0. This number is $(n-t)\binom{n}{t}$ because $|\Gamma_t| = \binom{n}{t}$ and each vertex in Γ_t is connected to $n - t$ vertices in Γ_{t+1}. Thus, $c(D_0) = \frac{(n-t)\binom{n}{t}}{n2^{n-1}} = \frac{\binom{n-1}{t}}{2^{n-1}}$. It remains to show that there is no system $D = (n, t, f)$ of lower cost.

We show this using network flow theory. We orient the edges of he hypercube from Γ_i to Γ_{i+1}, and we assign a capacity 1 to each of these arcs. An output function f induces a cut, i.e., an

(S,T)-partition of the vertices of the hypercube. In fact, S are the vertices that output 0 and T are the vertices that output 1. Since $\mathcal{N}^t(0^n) \subseteq S$ and $\mathcal{N}^t(1^n) \subseteq T$, we are in fact dealing with cuts separating Γ_t and Γ_{n-t}. The capacity of the cut is the number of edges starting in S and ending in T. This number (divided by $n2^{n-1}$) is precisely the average instability of D.

The cut induced by f_0 has capacity $(n-t)\binom{n}{t}$. If we prove that this is a minimum cut, we are done. We do this by describing a flow from Γ_t to Γ_{n-t} that saturates this cut, which proves the cut is minimum, by the min-cut/max-flow theorem. The flow on an arc linking a node of Γ_i to a node of Γ_{i+1} is uniformly $\frac{\binom{n-1}{t}}{\binom{n-1}{i}}$, for $t \leq i < n-t$. Notice that the law of flow conservation is satisfied since, at a vertex $x \in \Gamma_i$, $t < i < n-t$, one easily checks that the incoming flow $(n-i)\frac{\binom{n-1}{t}}{\binom{n-1}{i}}$ is equal to the outgoing flow $i\frac{\binom{n-1}{t}}{\binom{n-1}{i-1}}$. The flow of each arc e is at most 1, and the total transported flow is equal to $\binom{n}{i}(n-i)\frac{\binom{n-1}{t}}{\binom{n-1}{i}} = n\binom{n-1}{t}$. Finally $n\binom{n-1}{t} = (n-t)\binom{n}{t}$, which is equal to the capacity of the cut. □

3.3 Symmetric memoryless systems: the Ehrenfest urn model

Random walks turn out to be particularily interesting when the function f is symmetric, i.e., when it depends only on the distribution of 0's and 1's in the input vector. These functions are of the follwing type: "if i sensors are measuring the value 1 then the output is d". In these cases, we can *project* the hypercube into a path, whose vertices are $\{0,\ldots,n\}$. We can therefore assume that f is defined over this set of vertices, instead of over the vertices of the hypercube. Instead of considering the state x we consider the state $i = \#_1(x)$. We get a new Markov chain with transitions:

$$\mathbb{P}\{i \to (i+1)\} = \frac{n-i}{n} \qquad \mathbb{P}\{i \to (i-1)\} = \frac{i}{n}.$$

This process is known as the Ehrenfest urn model [7]. The probability of moving from i to $i+1$ is simply the probability of moving from a vertex with i entries equal to 1 to a vertex with $i+1$ entries equal to 1. There are $n-i$ such edges that can cause this to happen.

The invariant distribution of the Ehrenfest model (easily computable by projection of the uniform distribution of the hypercube) is known to be $\pi_i = \frac{\binom{n}{i}}{2^n}$. Therefore, the invariant distribution of the coupled states corresponding to the arcs $(i,i+1)$ and $(i+1,i)$ are $\pi_{(i,i+1)} = \frac{\binom{n}{i}}{2^n}\frac{n-i}{n}$, and $\pi_{(i+1,i)} = \frac{\binom{n}{i+1}}{2^n}\frac{i+1}{n}$. Thus, $\pi_{(i,i+1)} = \pi_{(i+1,i)} = \frac{\binom{n-1}{i}}{2^n}$.

Theorem 2. *Let $D = (n,t,f)$ be a symmetric memoryless system. Then,*

$$c(D) = \sum_{i=0}^{n-1} \binom{n-1}{i} \frac{|f(i+1) - f(i)|}{2^{n-1}}.$$

Proof. We know from the ergodic theorem that

$$c_{\lambda,l}(D) \underset{l\to\infty}{\longrightarrow} \sum_{i=0}^{n-1} \pi_{(i,i+1)}|f(i+1)-f(i)| + \sum_{i=0}^{n-1} \pi_{(i+1,i)}|f(i+1)-f(i)| \quad \text{a.s.}$$

The result follows from the fact that $\pi_{(i,i+1)} = \pi_{(i+1,i)} = \frac{\binom{n-1}{i}}{2^n}$. □

Remark 1. *We can recompute the instability of sytem D_0 introduced in previous section, because f_0 is symmetric: $f_0(i) = 1 \iff i > t$. Therefore, $c(D_0) = \binom{n-1}{t}\frac{1}{2^{n-1}}$.*

Remark 2. *D_0, in the context of the geodesic analysis in [5], appeared to be optimal only among the* symmetric *memoryless systems.*

4 Average instability of systems with memory

Let us define the following system having only one bit of memory: $D_1 = (n, t, S_1, f_1, \tau_1)$, where $S_1 = \{\perp, 0, 1\}$ and

$$f_1(x,s) = \tau_1(x,s) = \begin{cases} 0 & \text{if } \#_1(x) \leq t \\ 1 & \text{if } \#_0(x) \leq t \\ maj(x) & \text{if } s = \perp \\ s & \text{otherwise} \end{cases}$$

Here, $maj(x)$ returns the most common bit value in $x \in \{0,1\}^n$, and 1 in case of a tie.

Remark 3. *System D_1 is optimal in the geodesic cost model [5]. Nevertheless, the proof is much more complicated than in the memoryless case.*

Theorem 3. *For every system $D = (n, t, S, \tau, f)$, we have $c(D_1) \leq c(D)$. Moreover,*

$$c(D_1) = \left(2^{n-1} \sum_{k=t}^{n-t-1} \frac{1}{\binom{n-1}{k}}\right)^{-1}$$

and $\frac{1}{c(D_1)}$ is the average time necessary to pass from t to $n-t$ in the Ehrenfest model.

Proof. Let $D = (n, t, S, \tau, f)$ be an arbitrary system. Let $x_0, x_1, \ldots, x_l$ be the first $l+1$ vertices of a walk in the hypercube. We can associate to each of these vertices a symbol in $\{*, 0, 1\}$ depending on whether the system is not forced to decide ($t < \#_0(x_i), \#_1(x_i) < n - t$), it is forced to decide a 0 ($\#_1(x_i) \leq t$) or it is forced to decide a 1 ($\#_0(x_i) \leq t$). We therefore obtain a sequence $y \in \{*, 0, 1\}^{l+1}$. Let us delete the symbols $*$ in y in order to obtain a shorter binary string y'. Let $c'(y')$ be the number of changes in two consecutive symbols of y' (either 01 or 10). It is clear that the number of output changes of f over the same walk is at least $c'(y')$. Since $c'(y')$ is exactly the number of output changes of f_1 over the walk, we have that $c_{\lambda,l}(D_1) \leq c_{\lambda,l}(D)$ over each random walk. Therefore, $c(D_1) \leq c(D)$.

It remains to study the average instability of D_1. Since f_1 is symmetric, we can project the system into states of the form

$$(i,s) \in (\{0, \ldots, t-1\} \times \{0\}) \cup (\{t, \ldots, n-t\} \times \{0,1\}) \cup (\{n-t+1, \ldots, n\} \times \{1\}),$$

where i denotes the number of 1's the processors are reading and s denotes the last output of f_1 (we are not considering the case $s = \perp$ because it appears only once, at the beginning). The transitions are the following:

$$\mathbb{P}\{(i,s) \to (i+1, s')\} = (\frac{n-i}{n})\mathbf{1}_{\{f(i,s)=s'\}} \quad \mathbb{P}\{(i,s) \to (i-1, s')\} = (\frac{i}{n})\mathbf{1}_{\{f(i,s)=s'\}}$$

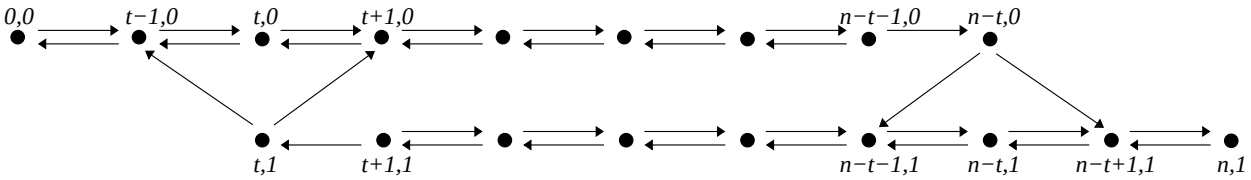

Figure 1: The auxiliar Markov chain for D_1 (with $t = 2$).

Previous Markov chain is irreducible and it has a stationary distribution π (to see that we only need to exhibit a positively recurrent state, for instance $(0, 0)$). We know that $c_{\lambda,l}(D_1)$ corresponds to the fraction of time the chain has been in states $(t, 1)$ or $(n-1, t)$ at time l. In fact, the system changes its decision each time it reads t 1's while its last decision was 1, or when it reads t 0's while its last decision was 0 (see Figure 1).

Then, by the ergodic theorem, $c_{\lambda,l}(D_1)$ converges almost surely to $\pi_{t,1} + \pi_{n-t,0}$. By symmetry, it follows that $c(D_1) = 2\pi_{t,1}$. We denote the expected time necessary to pass from a state (i, a) to a state (j, b) by $\mathbb{E}_{(i,a)}(T_{(j,b)})$. Since the inverses of the expected return times correspond to the stationary distribution, we have $c(D_1) = \frac{2}{\mathbb{E}_{(t,1)}(T_{(t,1)})}$. But, since each cycle passing through $(t, 1)$ also passes through $(n-t, 0)$,

$$\mathbb{E}_{(t,1)}(T_{(t,1)}) = \mathbb{E}_{(t,1)}(T_{(n-t,0)}) + \mathbb{E}_{(n-t,0)}(T_{(t,1)})$$

By symmetry, we have $\mathbb{E}_{(t,1)}(T_{(n-t,0)}) = \mathbb{E}_{(n-t,0)}(T_{(t,1)})$. On the other hand, for each $i \neq t$, we have $\mathbb{E}_{(t,1)}(T_{(i,0)}) = \mathbb{E}_{(t,0)}(T_{(i,0)})$. Thus we obtain:

$$c(D_1) = \frac{2}{2\mathbb{E}_{(t,0)}(T_{(n-t,0)})} = \frac{1}{\mathbb{E}_{(t,0)}(T_{(n-t,0)})}$$

The expected time to reach state $(n-t, 0)$ starting from $(t, 0)$ is equal to the expected time to reach state $n-t$ starting from state t in the classical Ehrenfest model. In fact, in this part of the walk the system will always be in state 0 and it only shifts to 1 when leaving the position $n-t$.

The last result follows from a theorem of [14], in which the authors prove that the expected time to go from state k to state $k+1$ in the Ehrenfest model is equal to $\frac{1}{\binom{n-1}{k}} \sum_{j=0}^{k} \binom{n}{j}$. Thus,

$$\frac{1}{c(D_1)} = \sum_{k=t}^{n-t-1} \frac{1}{\binom{n-1}{k}} \sum_{j=0}^{k} \binom{n}{j}$$

Stating $k' = n-1-k$ we get:

$$\frac{1}{c(D_1)} = \sum_{k'=t}^{n-t-1} \frac{1}{\binom{n-1}{n-1-k'}} \sum_{j=0}^{n-k'-1} \binom{n}{j} = \sum_{k'=t}^{n-t-1} \frac{1}{\binom{n-1}{k'}} \sum_{j=k'+1}^{n} \binom{n}{j}$$

Thus, adding the two expressions of $\frac{1}{c(D_1)}$ we have:

$$\frac{2}{c(D_1)} = \sum_{k=t}^{n-t-1} \frac{2^n}{\binom{n-1}{k}}.$$

□

5 The stabilizing role played by memory

In order to quantify the stabilizing role played by memory we must compute the number of decision changes of the optimal memoryless system D_0 *per each* decision change performed by the optimal 1-bit of memory system D_1. More precisely, we must study the ratio $\frac{c(D_0)}{c(D_1)}$. Since we are interested in the asymptotic behavior $n \to \infty$, we note n instead of $n-1$:

$$c_n(t) = \frac{c(D_0)}{c(D_1)} = \sum_{k=t}^{n-t} \frac{\binom{n}{t}}{\binom{n}{k}}$$

5.1 Upper bounds

First we prove an upper bound that is independent of t.

Lemma 1. $c_n(t) = O(\sqrt{n})$

Proof. Let us assume (w.l.o.g.) that n is even.

$$c_n(t) = 2\sum_{k=t}^{n/2} \frac{\binom{n}{t}}{\binom{n}{k}} = 2\sum_{k=t}^{n/2} \frac{(t+1)(t+2)\dots k}{(n-k+1)(n-k+2)\dots(n-t)}.$$

Let $1 \le s_n \le n/2$ (s_n grows with n; later it becomes clear why we should choose $s_n = \sqrt{n}$). If $t+s_n > n/2$ then $c_n(t) \le 2\sum_{k=n/2-s_n+1}^{n/2} 1 = 2s_n$. Let us consider the case $t+s_n \le n/2$. It follows:

$$\begin{aligned}
c_n(t) &= 2\sum_{k=t}^{t+s_n} \frac{(t+1)(t+2)\dots k}{(n-k+1)(n-k+2)\dots(n-t)} + 2\sum_{k=t+s_n+1}^{n/2} \frac{(t+1)(t+2)\dots k}{(n-k+1)(n-k+2)\dots(n-t)} \\
&\le 2 + 2s_n + 2\sum_{k=t+s_n+1}^{n/2} \frac{(t+1)\dots(t+s_n)}{(n-t)\dots(n-t-s_n+1)} \quad \frac{(t+s_n+1)\dots k}{(n-t-s_n)\dots(n-k+1)} \\
&\le 2 + 2s_n + 2\sum_{k=t+s_n+1}^{n/2} \frac{(t+1)\dots(t+s_n)}{(n-t)\dots(n-t-s_n+1)} \le 2 + 2s_n + 2(n/2-t-s_n)\left(\frac{t+s_n}{n-t-s_n}\right)^{s_n}
\end{aligned}$$

Let $x = n/2 - t - s_n$. Let $g_n(x) = 2 + 2s_n + 2x\left(\frac{1-\frac{2x}{n}}{1+\frac{2x}{n}}\right)^{s_n}$. Since $c_n(t) \le g_n(x)$, our goal is to find the maximum of $g_n(x)$. By computing $g_n'(x_0) = 0$ we get

$$x_0 = \frac{ns_n}{2}\left(-1 + \sqrt{1+\frac{1}{s_n^2}}\right) = \frac{ns_n}{2}\left(-1+1+\frac{1}{2s_n^2} + o\left(\frac{1}{s_n^2}\right)\right) = \frac{n}{4s_n} + o\left(\frac{n}{s_n}\right).$$

Since $\frac{2x_0}{n} = \frac{1}{2s_n} + o\left(\frac{1}{s_n}\right)$, it follows:

$$\begin{aligned} c_n(t) &\leq 2 + 2s_n + 2\left(\frac{n}{4s_n} + o\left(\frac{1}{s_n}\right)\right) \exp^{s_n(\log(1-\frac{1}{2s_n}+o\left(\frac{1}{s_n}\right))-\log(1+\frac{1}{2s_n}+o\left(\frac{1}{s_n}\right))} \\ &\leq 2 + 2s_n + 2\left(\frac{n}{4s_n} + o\left(\frac{1}{s_n}\right)\right) \exp^{s_n(-\frac{1}{2s_n}-\frac{1}{2s_n}+o\left(\frac{1}{s_n}\right))} \\ &\leq 2 + 2s_n + 2\left(\frac{n}{4s_n} + o\left(\frac{1}{s_n}\right)\right) \exp^{-1+o(1)} = O(s_n + \frac{n}{s_n}) \end{aligned}$$

Since either s_n or $\frac{n}{s_n}$ grows faster than $\sqrt{n}$, the best we can do is to choose $s_n = \sqrt{n}$ in order to conclude $c_n(t) = O(\sqrt{n})$.

□

5.2 Lower bounds

The main question is whether there are values of t for which the previous upper bound is achieved, i.e., whether $c_n(t) = \Theta(\sqrt{n})$ for some relation between n and t. We are going to see first that this does not happen for extremes values of t.

Lemma 2. *When t is constant, $\lim_{n\to\infty} c_n(t) = 2$.*

Proof. Notice first that $c_n(t) = 2 + \sum_{k=t+1}^{n-t-1} \frac{\binom{n}{t}}{\binom{n}{k}}$. When t is a constant, $\frac{\binom{n}{t}}{\binom{n}{t+1}} = \frac{\binom{n}{t}}{\binom{n}{n-t-1}} = O(1/n)$. For $t+2 \leq k \leq n-t-2$, $\frac{\binom{n}{t}}{\binom{n}{k}} = O(1/n^2)$ and there are $O(n)$ such k's. Thus, $\lim_{n\to\infty} c_n(t) = 2$. □

The previous result can be explained as follows: for t constant, the Markov process will rarely enter a state where it needs to change its decision, either with D_0 or D_1. Thus, the chain, between two forced decisions, will be shuffled. Once shuffled, in half of the cases the chain will force the system to take the same decision it took before. On the other hand, when t is close to $n/2$, almost all the decisions are forced. In this case the ratio is also bounded by a constant. In fact,

Lemma 3. *When $t = n/2 - \beta$, $\lim_{n\to\infty} c_n(t) \leq 2 + 2\beta$.*

Proof. We have $c_n(t) = 2 + \sum_{k=t+1}^{n-t-1} \frac{\binom{n}{t}}{\binom{n}{k}}$. The sum above contains (less than) 2β terms, which are all smaller than 1. Thus, $c_n(t) \leq 2 + 2\beta$ when n tends to infinity. □

The interesting case appears when t is close to $n/2$ but this distance grows with n. More precisely,

Lemma 4. *Let $t = n/2 - \alpha\sqrt{n}$ (with α constant). Then $c_n(t) = \Theta(\sqrt{n})$.*

Proof. It only remains to prove the lower bound (see Lemma 1).

$$
\begin{aligned}
c_n(n/2-\alpha\sqrt{n}) &\geq \sum_{k=0}^{\alpha\sqrt{n}} \frac{\binom{n}{n/2-\alpha\sqrt{n}}}{\binom{n}{n/2-\alpha\sqrt{n}+k}} \\
&= \sum_{k=0}^{\alpha\sqrt{n}} \frac{(n/2-\alpha\sqrt{n}+1)(n/2-\alpha\sqrt{n}+2)\ldots(n/2-\alpha\sqrt{n}+k)}{(n/2+\alpha\sqrt{n})(n/2+\alpha\sqrt{n}-1)\ldots(n/2+\alpha\sqrt{n}-k+1)} \\
&\geq \sum_{k=0}^{\alpha\sqrt{n}} \left(\frac{n/2-\alpha\sqrt{n}}{n/2+\alpha\sqrt{n}}\right)^k \geq \alpha\sqrt{n}\left(\frac{1-\frac{2\alpha}{\sqrt{n}}}{1+\frac{2\alpha}{\sqrt{n}}}\right)^{\alpha\sqrt{n}} \\
&= \alpha\sqrt{n}\exp^{\alpha\sqrt{n}(\log(1-\frac{2\alpha}{\sqrt{n}})-\log(1+\frac{2\alpha}{\sqrt{n}}))} = \alpha\sqrt{n}\exp^{\alpha\sqrt{n}(1-\frac{2\alpha}{\sqrt{n}}-1-\frac{2\alpha}{\sqrt{n}}+o(\frac{1}{\sqrt{n}}))} \\
&= \alpha\sqrt{n}\exp^{-4\alpha+o(\frac{1}{\sqrt{n}})} = \Omega(\sqrt{n})
\end{aligned}
$$

□

References

[1] P. Berman and J. Garay, "Cloture votes: $n/4$-resilient distributed consensus in $t+1$ rounds," *Math. Sys. Theory,* 26(1): 3-19, 1993.

[2] A.P. Chandrakasan and R.W. Brodersen, *Low power digital CMOS design,* Kluwer Academic Publishers, 1995.

[3] L. Davidovitch, S. Dolev and S. Rajsbaum, "Stability of Multi-Valued Continuous Consensus," *SIAM J. on Computing,* 37(4), 1057-1076 (2007). Extended abstract appeared as "Consensus Continue? Stability of Multi-Valued Continuous Consensus!", *6th Workshop on Geometric and Topological Methods in Concurrency and Distributed Computing*, (GETCO 2004), October 2004.

[4] S. Dolev, *Self-Stabilization*, the MIT Press, March 2000.

[5] Shlomi Dolev, and Sergio Rajsbaum, "Stability of Long-lived Consensus" *J. of Computer and System Sciences*, Vol. 67, Issue 1, pp. 26-45, August 2003. Preliminary version in *Proc. of the 19th Annual ACM Symp. on Principles of Distributed Computing*, (PODC 2000), pp. 309–318, 2000.

[6] Persi Diaconis, Mehrdad Shahshahani, "Generating a random permutation with random transpositions," *Probability Theory and Related Fields,* Volume 57, Number 2, pp. 159–179, June 1981.

[7] Paul Ehrenfest and Tatiana Ehrenfest, "Ueber zwei bekannte Eingewände gegen das Boltzmannsche H-Theorem," *Zeitschrift für Physik* 8, pp. 311–314, 1907.

[8] J. Kahn, G. Kalai and N. Linial, "The Influence of Variables on Boolean Functions," *Proc. of the IEEE FOCS*, 68–80, 1988.

[9] B. Kramer and M. Fussenegger, "Hysteresis in a synthetic mammalian gene network," *Proc. Natl Acad Sci USA* 102(27), 9517-9522 (2005).

[10] S. Kutten and T. Masuzawa, "Output Stability Versus Time Till Output," *Proc. DISC*, LNCS 4731, 343-357 (2007).

[11] H. Kopetz and P. Veríssimo, "Real Time and Dependability Concepts," chapter 16, pp.411-446 in Sape Mullender (ed.), *Distributed Systems,* ACM Press, 1993.

[12] E. Musoll, T. Lang, J. Cortadella, "Exploiting the locality of memory references to reduce the address bus energy," *Proc. of the Int. Symp. on Low Power Electronics and Design,* Aug. 1997, pp. 202–207.

[13] James R. Norris, *Markov Chains,* Cambridge Series in Statistical and Probabilistic Mathematics, Cambridge University Press, Oct. 1998.

[14] J.L. Palacios, "Another Look at the Ehrenfest Urn via Electric Networks," *Advances in Applied Probability,* Vol. 26, No. 3, pp. 820–824, Sept. 1994.

[15] J. Pomerening. E. Sontag and J. Ferrell, "Building a cell cycle oscillator: hysteresis and bistability in the activation of Cdc2," *Nature Cell Biology* 5, 346 - 351 (2003).

[16] C-L. Su, C-Y. Tsui, A.M. Despain, "Saving power in the control path of embedded processors," *IEEE Design & Test of Comp.,* pp. 24–30, 1994.