How to Play the Majority Game with Liars

Steve Butler[1], Jia Mao[2⋆], and Ron Graham[2⋆⋆]

[1] Dept. of Mathematics, University of California, San Diego,
La Jolla, CA 92093-0112 sbutler@math.ucsd.edu
[2] Dept. of Computer Science and Engineering, University of California, San Diego,
La Jolla, CA 92093-0404 jiamao@cs.ucsd.edu, graham@ucsd.edu

Abstract. The *Majority* game is a two player game with a questioner **Q** and an answerer **A**. The answerer holds n elements, each of which can be labeled as 0 or 1. The questioner can ask questions comparing whether two elements have the same or different label. The goal for the questioner is to ask as few questions as possible to be able to identify a single element which has a majority label, or in the case of a tie claim there is none. We denote the minimum number of questions **Q** needs to make, regardless of **A**'s answers, as q^*. In this paper we consider a variation of the Majority game where **A** is allowed to lie up to t times, i.e., **Q** needs to find an *error-tolerant* strategy. In this paper we will give upper and lower bounds for q^* for an adaptive game (where questions are processed one at a time), and will find q^* for an oblivious game (where questions are asked in one batch).

1 Introduction

The well-studied *Majority* game consists of two players: a questioner **Q** and an answerer **A**. Initially **A** holds a set of n elements, each of which will have a binary label (e.g., 0 or 1), and **Q** asks questions as to whether two elements have the same or different labelling. In the game, **Q**'s goal is to identify one element of the majority label (or in the case of a tie, claim that there is none), while **A**'s goal is to block **Q** from identifying such an element. If after no more than q questions **Q** can identify a majority element then **Q** wins, otherwise **A** wins. We say **Q** has a *winning strategy* of length q if **Q** can always win the game with at most q questions, regardless of what **A** answers. The goal is to design strategies for **Q** with minimal q, denoted by q^*.

1.1 History

The earliest variant of the *Majority* problem was originally proposed by Moore in the context of fault-tolerant system design in 1981 [11]. A number of different variants were subsequently proposed and analyzed. This problem resurfaced after about twenty years in a military application where communication needs to be

⋆ Partially supported by an NSF graduate fellowship
⋆⋆ Research partially supported by NSF Grant CCR-0310991

minimized to locate one sensor that has not been corrupted among a group of sensors [6].

There have been several variations of the majority game studied in past literature. Variations have included examining different k (the number of permissible labels); considering *adaptive* or *oblivious* versions of the game (in an adaptive game, **Q** learns the answer to each question before asking the next question, while in an oblivious game, **Q** asks all questions in one batch before getting any answers from **A**); and whether or not a majority label is known to exist or not. (See [1, 3, 6, 7, 9, 14, 17].)

In the adaptive case with 2 labels, Saks and Werman [14] were the first to prove a tight bound of the minimum length of a winning strategy for **Q** to be $n - \mu_2(n)$, where $\mu_2(n)$ is the number of 1s in n's binary expansion. Different proofs for the same result were subsequently given in [3] and [17]. When k is unknown, a tight bound of $\lceil 3n/2 \rceil - 2$ was given in [9]. The average case of the same setting was analyzed in [4]. Similar bounds were proven for randomized algorithms [10].

In the oblivious case, when k is unknown, the optimal winning strategy for **Q** is much harder to design or analyze. If the existence of a majority label is not known a priori, **Q** needs $\Omega(n^2)$ many questions [15]. However, if a majority label is known to exist, by using a special type of graph, called Ramanujan graph, there is a constructive strategy for **Q** that uses no more than $(1 + o(1))27n$ queries. The constant 27 can be further improved to 19.5 if only existence of such a strategy is desired [7].

1.2 Error tolerance

In past literature, **A** is always a *malevolent but truthful* adversary in the sense that as long as their current answer is *consistent* with previously given answers, they will want to win the game. However, an *error-tolerant* feature is desired when the answers to the queries in the application may be faulty due to communication errors. In this paper we address this issue by putting the *Majority* game in a broader context of fault-tolerant communication, namely, searching games with errors. Generally, these games are more difficult to analyze, but have a much wider range of applications compared to perfect information two person games. One such famous game is the *Rényi-Ulam liar game* [13, 16]. For a comprehensive overview of this topic, we refer the reader to a recent survey [12].

In this paper we consider bounded error tolerance for the *Majority* game, where **A** is allowed to *lie up to a fixed number t times*. More precisely, this means that after **Q** specifies an element as being in the majority or state there is none, **A** has the freedom to flip up to t answers of the previously asked queries and reveal a labelling that is consistent with this modified set of answers. Now that **A** can lie how much will this handicap **Q**? What is the new minimal q^* and what strategy should **Q** adopt to achieve this bound?

In this paper we will begin to answer some of these questions for the *Majority* game with binary labels. We will give upper and lower bounds for q^* by producing strategies for **Q** and **A** in various versions of the game. For the sake of exposition,

some of the justification for these strategies will be omitted from this paper and will appear in a longer version of this paper. We summarize our results in the table below, where t is the number of lies and n is the number of objects.

Game	n is odd	n is even
Adaptive $t = 1$	$n \le q^* \le n+1$	$n+1 \le q^* \le n+2$
Adaptive $t > 1$	$\lceil \frac{t+3}{4}n - \frac{t+1}{4} \rceil \le q^* \le \big(\frac{t+1}{2} + o(1)\big)n$	$\frac{t+1}{2}n \le q^* \le \big(\frac{t+1}{2} + o(1)\big)n$
Oblivious $t \ge 1$	$q^* = \lceil (t + \frac{1}{2})n \rceil$	

The upper bound in the adaptive case for $t > 1$ holds only when $t = o(n^{1/2})$. A more precise statement of the upper bound is given by Theorem 4. For the case when t is large in comparison to n then a better upper bound might be given by the oblivious upper bound, this of course is dependent on the size of t.

Also note that there is a difference between the case when n is odd and n is even. This is due to the fact that when n is odd there *must* be a majority element which gives additional information **Q** can use in forming a strategy.

2 Adaptive Setting

We can adapt the known strategy for the game with no lies to give a simple upper bound for q^*. Recall that for the majority game of binary labels with no lies the optimal winning strategy takes $(n - \mu_2(n))$ queries [14].

Theorem 1. *In the adaptive Majority game on n elements with binary labels and at most t lies,*

$$q^* \le (t+1)\big(n - \mu_2(n)\big) + t$$

where $\mu_2(n)$ is the number of 1s in n's binary expansion.

Proof. Let **Q** ask the same queries as in the best strategy to play the *Majority* game with no lies, only that each query is repeated until $(t+1)$ answers agree (at which point we know the relationship between the two elements) before going to the next query. Because **A** is not allowed to lie more than t times, the total number of queries **Q** needs to ask is $(t+1)\big(n - \mu_2(n)\big)$ plus at most t. □

This simple bound for q^* can be improved by **Q** using error detection and correction from the answers of **A**.

2.1 Preliminaries

To keep track of the game as it progresses, we use an auxiliary (multi-)graph. The n objects are the vertices and edges correspond to comparisons between elements. (We will allow queries to be repeated and any multiple queries are

represented by multi-edges.) As the game progresses **Q** will give **A** an edge and ask **A** to color it blue if the two elements have the same label and red it the two elements have different labels.

One of the most important tools used in the formation of a strategy is to use the auxiliary graph to detect lies. For instance, note that if **A** were truthful then every cycle would always have an even number of red edges. But when **A** can lie this no longer needs to be the case and we have the following observation.

Observation 1. *A cycle can have an odd number of red edges if and only if an odd number of edges in the cycles correspond to lies.*

This follows by noting that the number of red edges corresponds to the number of times along the cycle that we switch labels. Initially if there were no lies we would have to switch an even number of times (i.e., we must return to the same label we started with). Now for every lie we either increase or decrease by 1 the number of switches made.

From the observation any cycle which contains an odd number of red edges must contain a lie, we will refer to such cycles as invalid, otherwise we say that the cycle is valid. It is important to notice though that just because a cycle is valid it does not imply that there are no lies, only that there are an even number of them.

2.2 Majority Game with at most $t = 1$ lie

Theorem 1 gives an upper bound of $\big(2(n-\mu_2(n))+1\big)$ for **Q**'s adaptive strategy when **A** is allowed to lie once. This bound can be improved by using *validity checking* of the auxiliary graph. (For the case involving one lie we have that a cycle is valid if and only if it has no lies, in general this will not hold.)

Theorem 2. *In the adaptive Majority game on n elements with binary labels and at most 1 lie*

$$q^* \leq \begin{cases} n+1 & \text{if } n \text{ is odd,} \\ n+2 & \text{if } n \text{ is even.} \end{cases}$$

Proof. We give a two-stage strategy for **Q** satisfying the bounds.
Stage 1: In the first stage **Q** starts by growing long blue paths by the following rule: connect the ends of two blue paths which have an equal number of vertices. This continues until either there are no two paths with the same number of vertices or we get a red edge.

In the first case **Q** takes the longest blue path and closes it up by a single question, if the cycle is blue then **Q** identifies any element on the cycle as a majority element (i.e., in such a case it is easy to see that the cycle contains more than half of the elements and all of them must have the same label), otherwise if the edge is red (i.e., the cycle is invalid so contains a lie) **Q** goes to the second stage.

In the second case there is a path with one single red edge in the middle. **Q** then asks a single question to close it up to form a cycle. If the new edge is red,

then the coloring is valid and so the cycle has an equal number of each label, **Q** then removes this component from the graph and continues as before. If the new edge is blue (i.e., the cycle is invalid so contains a lie) **Q** goes to the second stage.

Stage 2: Starting stage 2 we already know that **A** has used their lie and so all subsequent answers must be true. We initially have one cycle with a single red edge (denoted by $\{u, v\}$) and possibly several blue paths. The first step in this stage is to connect one vertex from each blue path to u. The graph now consists of a cycle with a tree attached to u. At this point **Q** has asked exactly n questions.

Because the lie lies in the cycle, all edges involved in the tree reflect truthful responses. In particular, **Q** can determine how many of the vertices of the cycle must have the same label as u in order for u to be the majority element, denote this number by k. Starting at u and going in the opposite direction of v, **Q** counts out k vertices. Denote the kth vertex by w. (If the cycle contains fewer than k vertices then u is in the minority and it will be easy to identify a majority element in the tree. Similarly, if $k \leq 0$ then u has a majority label.)

Q now queries the edge $\{u, w\}$. If the edge is blue then all the vertices between u and w have the needed label and we can conclude that u is a majority element. On the other hand if the edge is red then there is a lie somewhere between u and w and so u cannot be a majority element. In the case when n is even we then only need to compare u with the vertex that precedes w to test if there is a tie. If the edge is blue then there is a tie, while if the edge is red then w is a majority element.

The result now follows by counting the number of queries used. □

To find a lower bound for q^* we need to give a strategy for **A**. Since **A** can adopt the same strategy as in the game with no lies we have that $q^* \geq n - \mu_2(n)$. However, **A** can do better as shown in the next theorem.

Theorem 3. *In the adaptive Majority game on n elements with binary labels and at most 1 lie*

$$q^* \geq \begin{cases} n & \text{if } n \text{ is odd,} \\ n+1 & \text{if } n \text{ is even.} \end{cases}$$

The case for n is odd will follow from Theorem 5 with $t = 1$. The proof for n is even will be found in the longer form of this paper.

2.3 Majority Game with at most $t \geq 2$ lies

We now consider the game where **A** is allowed to lie up to $t \geq 2$ times. We first start by establishing an upper bound.

Theorem 4. *In the adaptive Majority game on n elements with binary labels and at most $t \geq 2$ lies,*

$$q^* \leq \frac{t+1}{2}n + 6t^2 + 2t + 3\log n.$$

Proof. We give a sketch of the strategy here. More details and justification can be found in the longer form of this paper. The strategy for **Q** will be to use two rounds. The first round will be "oblivious" in that **Q** will always ask the same set of questions (this round will use $(t+1)n/2$ questions). In the second round **Q** then uses the answers from the first round to find and correct all lies.

We first consider the case $n > 2t$ with n even.

In the first stage **Q** forms an n-cycle with the n vertices and asks $\lfloor t/2 \rfloor$ questions on each edge of the cycle. **Q** then makes

$$t + 1 - 2\left\lfloor \frac{t}{2} \right\rfloor = \begin{cases} 1 & \text{if } t \text{ is even,} \\ 2 & \text{if } t \text{ is odd,} \end{cases}$$

queries between opposite vertices of the cycles (we will refer to these queries as spokes). An example is shown in Figure 1.

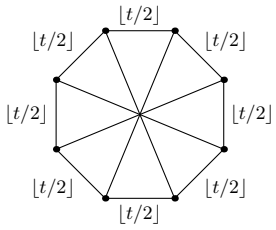

Fig. 1. "Oblivious" first round queries.

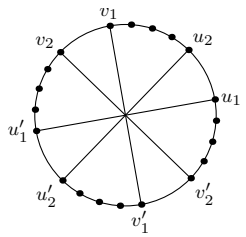

Fig. 2. Looking for lies in the spokes.

This strategy has the following useful property. If **A** has lied then there is either an invalid 2-cycle either in the spokes or along the exterior; or there is an invalid 4-cycle of the form $v_1v_1'v_2'v_2$ (see Figure 2); or there is an invalid $n/2$ cycle of the form $v_1v_1'v_2'\ldots v_1$ (i.e., lying along half of the outer cycle plus a spoke).

We can quickly find and remove all errors from invalid 2-cycles and 4-cycles. Namely, for each such cycle we make $2t$ queries on each of 3 edges and take the majority answer on each edge, if we have not yet found the lie from this then the remaining edge is a lie and we then can correct. Thus we would need at most $6t^2$ queries to correct these lies.

If after correcting these queries there is still an invalid $n/2$ cycle then it must be the case that there are two opposite intervals "saturated" with lies (i.e., all queries between u_1u_2 and $u_1'u_2'$ in Figure 2 are lies). In particular, there can only be at most one lie left. We now lift up the $n/2$ cycle (which will have only one lie) and locate the lie by using a splitting technique. Namely we join two opposite pairs of vertices with three edges and take the majority answer and use this to split the cycle into two smaller cycles, one of which will be invalid (and which contains the lie of the $n/2$ cycle). We then continue this process of cutting in half each time until we have located the lie. In particular, this technique will locate the lie in $3\log n$ steps.

Q is now finished because they can detect and correct lies given by **A** and relate all elements together. In particular, **Q** has used at most $(t+1)n/2+6t^2+3\log n$ queries to accomplish this.

For the case n odd **Q** sets aside a single element and runs the procedure and then at the end connects the element back into the graph by making at most $2t+1$ queries relating the odd element out with some arbitrary element.

Finally for the case $n \leq 2t$ we can simply build a tree where we keep asking questions on each edge until we get $t+1$ responses which agree. In particular, we would need at most $2t^2+t$ queries in such a case.

Putting it all together gives the desired result. □

To establish the lower bound, we will make use of the following general observation.

Observation 2. *If the coloring is valid (i.e., no lies are detected), then* **Q** *will not be able to determine the correct relationship for an element which is involved in no more than* t *queries.*

This observation follows by noting that since the coloring is valid and **A** is allowed to change the color of up to t edges, then **A** can change all the queries involved with a vertex of low degree (i.e., no more than t) and still produce an admissible labelling.

Theorem 5. *In the adaptive majority game on* n *elements with binary labels and at most* $t \geq 1$ *lies,*

$$q^* \geq \begin{cases} \dfrac{t+1}{2}n & \text{if } n \text{ is even,} \\ \left\lceil \dfrac{t+3}{4}n - \dfrac{t+1}{4} \right\rceil & \text{if } n \text{ is odd.} \end{cases}$$

Proof. For the case when n is even, **A** can use the following strategy: Initially assign half of the elements with label 0 and the other half with label 1 and answer all of the questions truthfully. If **Q** makes fewer than $(\frac{t+1}{2})n$ queries, then by degree considerations there is a vertex with degree at most t. Based on the above observation, **Q** will be unable to determine the correct relationship of that vertex with the remaining elements and in particular will not be able to distinguish between a tie and the existence of a majority element.

The case for n is odd will be found in the longer form of this paper. □

3 Oblivious Setting

In the oblivious setting, **Q** has to specify all the edges in the auxiliary graph G before **A** colors any of them. This implies that **Q** has to accomplish *detection* and *location* of lies simultaneously. We have another important observation.

Observation 3. *In the Majority game of binary labels with at most t lies, if an edge e is part of $2t$ cycles that pairwise edge-intersect only at e (though they might share many vertices in common), then a lie is located at e if and only if at least $(t+1)$ of these cycles are invalid.*

This observation follows by noting that if an edge corresponds to a truthful answer then there can be at most t of the $2t$ cycles intersecting at e which can be invalid. On the other hand if the edge corresponded to a lie then at most $t-1$ of the $2t$ cycles intersecting at e can be valid, or equivalently, at least $t+1$ invalid cycles.

Theorem 6. *In the oblivious Majority game on n elements with binary labels and at most $t \geq 1$ lies,*

$$q^* = \left\lceil (t + \frac{1}{2})n \right\rceil.$$

Proof. We first establish the upper bound. The observation above implies that if we can construct an auxiliary graph for $\mathbf{Q}$ such that for any particular edge we can find $2t$ cycles that are pairwise edge-joint only at that edge, we can locate and thus correct all possible lies with no more queries needed.

We handle the base cases first. For $n = 2$, we use $(2t+1)$ edges for the same query. For $n = 3$, the query graph is a triangle with one query asked t times and the other two queries each asked $(t+1)$ times.

For even $n \geq 4$, we construct a multigraph as shown in Figure 3 where all edges in the outer cycle are multi-edges (repeated t times) and single edges (or spokes) connect each pair of opposite vertices. The total number of edges is therefore $(t+\frac{1}{2})n$. For odd $n \geq 3$, first construct a graph as in the $n+1$ case and then contract a set of edges on the outer cycle, an example is shown in Figure 4. In this case it can be checked that there are $\lceil (t+\frac{1}{2})n \rceil$ edges in the graph.

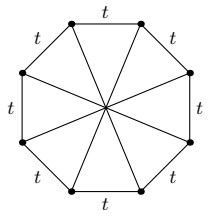

Fig. 3. Oblivious graph, n even.

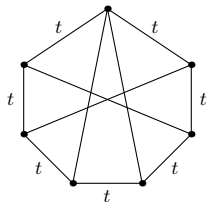

Fig. 4. Oblivious graph, n odd.

For each spoke, we can find t cycles using one half of the outer cycle and another t using the other half. For each outer edge e, first we can find $(t-1)$ small cycles by joining it with the other $(t-1)$ multiedges with the same endpoints,

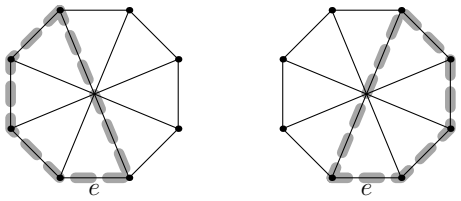

Fig. 5. The remaining two cycles for a side edge e.

then we use edges in the outer cycle to obtain another $(t-1)$ cycles. We need two more cycles and these are constructed using the spokes and the remaining unused edges of the outer cycle as shown in Figure 5. Because each edge lies in at least $2t$ cycles pairwise edge-joint only at that edge, all lies can be located and hence corrected. Establishing the upper bound.

For the lower bound, we again consider the degrees. If **Q** asks fewer than $\lceil (t+\frac{1}{2})n \rceil$ then since **Q**'s strategy is oblivious, **A** can examine the entire auxiliary graph and find a vertex v with degree at most $2t$. **A** can split the remaining vertices into two sets U and V as equally as possible (i.e., $\big||U|-|V|\big| \leq 1$) with label 0 and 1 respectively. For queries not involving v, **A** answers truthfully. For queries involving v, **A** answers half of the queries as if v is labelled 0 and the other half as if v is labelled 1. This is possible because **A** is allowed to lie up to t times. Now **Q** cannot distinguish which half are lies and hence cannot determine the label for v which is essential because the other vertices are in an (almost) exact balance. This establishes the lower bound and concludes the proof. □

4 Conclusion and Remarks

Motivated by the practical need of an *error-tolerant* feature, we have concentrated on optimizing **Q**'s strategy in the presence of lies (or errors) for binary labels in the *Majority* game. We point out that when the number of lies is upper bounded by a constant t, **Q** can still win the game with a linear (in n) number of questions. Upper and lower bounds on the length of **Q**'s optimal strategy were derived in both the adaptive setting and the oblivious setting.

Consideration of fault-tolerance may also be useful for the many other variants of the *Majority* game, such as when the number of different labels is more than two. A natural generalization of the Majority game is the *Plurality* game where **Q** wants to identify one element of the *plurality* label (most frequently occurring), still using only pairwise equal/unequal label comparisons of elements. Much attention has been given to designing adaptive strategies (deterministic or randomized) for fixed or unknown k (see [2, 5, 8, 10, 15]). We remark here that the same reasoning of Theorem 1 applies to existing bounds for all variants of the *Majority* game (including the *Plurality* game) if the maximum number of lies allowed t is fixed. The new upper bounds will only be at most worse by a multiplicative constant $(t+1)$ and an additive constant t.

In this paper, we gave a complete picture for the *oblivious* setting in the *Majority* game with a constant bounded number of lies. In the *adaptive* setting, however, there are still various gaps between the upper and lower bounds obtained. Closing these gaps would be interesting to pursue. In the meantime, other types of error-tolerance may also be considered, such as bounded error fraction or random errors.

5 Acknowledgement

We thank Joel Spencer, Robert Ellis and anonymous referees on an earlier version of this paper for helpful discussions and suggestions.

References

1. M. Aigner, "Variants of the majority problem", *Applied Discrete Mathematics* **137** (2004), 3–25.
2. M. Aigner, G. De Marco, M. Montangero, "The plurality problem with three colors and more", *Theoretical Computer Science* **337** (2005), 319–330.
3. L. Alonso, E. Reingold, R. Schott, "Determining the majority", *Information Processing Letters* **47** (1993), 253–255.
4. L. Alonso, E. Reingold, R. Schott, "Average-case complexity of determining the majority", *SIAM J. Computing* **26** (1997), 1–14.
5. L. Alonso, E. Reingold, "Determining plurality", *manuscript*, 2006.
6. F. R. K. Chung, R. L. Graham, J. Mao, and A. C. Yao, "Finding favorites", *Electronic Colloquium on Computational Complexity (ECCC)* (2003) 078, 15pp.
7. F. R. K. Chung, R. L. Graham, J. Mao, and A. C. Yao, "Oblivious and adaptive strategies for the majority and plurality problems", *COCOON 2005*, 329–338; *Algorithmica*, to appear.
8. Z. Dvorak, V. Jelinek, D. Kral, J. Kyncl, and M. Saks, "Three optimal algorithms for balls of three colors", *STACS 2005*, Lecture Notes in Comp. Sci. **3404**, Springer, Berlin, 2005, 206–217.
9. M. J. Fischer and S. L. Salzberg, "Finding a majority among n votes", *J. Algorithms* **3** (1982), 375–379.
10. D. Kral, J. Sgall, and T. Tichỳ, "Randomized strategies for the plurality problem", *manuscript*, 2005.
11. J. Moore, "Proposed problem 81-5", *J. Algorithms* **2** (1981), 208–210.
12. A. Pelc, "Searching games with errors — fifty years of coping with liars", *Theoret. Comput. Sci.* **270** (2002), 71–109.
13. A. Rényi, "On a problem in information theory", *Magyar Tud. Akad. Mat. Kutató Int. Közl.* **6** (1961), 505–516.
14. M. Saks and M. Werman, "On computing majority by comparisons", *Combinatorica* **11** (1991), 383–387.
15. N. Srivastava, and A. D. Taylor, "Tight bounds on plurality", *Information Processing Letters* **96** (2005), 93–95.
16. S. M. Ulam, **Adventures of a mathematician**, Charles Scribner's Sons, New York, 1976, xi+317pp.
17. G. Wiener, "Search for a majority element", *J. Statistical Planning and Inference* **100** (2002), 313–318.