Beeping an MIS

(Extended Abstract)

Yehuda Afek [*] Noga Alon [†] Ziv Bar-Joseph [‡]

Abstract

We consider an extremely harsh synchronous broadcast communication model and present two algorithms solving the Maximal Independent Set (MIS) problem in this model. The model consists of an anonymous broadcast network, in which nodes have no knowledge about the network topology nor any bound on the network size. Furthermore, nodes may start the algorithm spontaneously in an asynchronous manner. In this model we consider two weak communication models, beeping with collision detection and beeping without collision detection. We present a randomized MIS algorithm for the former model that works in $O(\log^2 n)$ rounds. For the latter we present an algorithm that produces an MIS in $O(\log^3 n)$ rounds but the nodes do not know that, and may continue to send messages indefinitely. Finally, we show that our algorithm is optimal under some restriction, by presenting a tight lower bound of $\Omega(\log^2 n)$ on the number of rounds required to construct a MIS for a restricted model.

Intended for Regular Presentation.

Contact Author:
Ziv Bar-Joseph
Email: `zivbj@cs.cmu.edu`

[*]The Blavatnik School of Computer Science, Tel-Aviv University, Israel 69978. afek@post.tau.ac.il

[†]School of Mathematics, & The Blavatnik School of Computer Science, Tel-Aviv University, Israel 69978. nogaa@post.tau.ac.il

[‡]School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213, USA. zivbj@cs.cmu.edu

1 Introduction

Motivated by biological observations on the mechanism by which a subset of cells in the fly's brain are selected to grow a bristle (stiff sensory hair) on the surface of the forehead we have recently suggested the network fly model for the solution of the maximal independent set (MIS) problem [1]. In that model there is a set of synchronous and anonymous processors communicating by local beeping [5] broadcasts in an arbitrary topology. Furthermore, in [1] we made the following three additional assumptions: that all the processors wake up together at the same synchronous round, that a bound on the network size is known to the processors, and collision detection. That is, processors can listen on the medium while broadcasting (as in some radio and local area networks).

In this paper we study the power of a weaker model to solve the MIS problem probabilistically. We present two new MIS algorithms, the first removes the synchronous wake-up and knowledge on network size assumptions, and the second removes in addition the collision detection assumption. The first algorithm constructs an MIS in $O(\log^2 n)$ rounds, where n is the number of nodes in the network, which is not known to the processors. The second algorithm constructs an MIS in $O(\log^3 n)$ rounds but since there is no collision detection and no knowledge whatsoever on the network size the algorithm does not terminate. Processors in the MIS have to beep forever in a certain probabilistic pattern, though an MIS might have been constructed already, but they do not know that. However, if, even a very rough estimate on the network size is given, then the algorithm can easily decide to terminate after $O(\log^3 N)$ rounds, where N is the estimate.

An MIS is a *maximal* set of nodes in a network such that no two of them are neighbors. Since the set is maximal every node in the network is either in the MIS or a neighbor of a node in the MIS. The problem of distributively selecting an MIS has been extensively studied in various models [2, 4, 22, 14, 15, 16, 17, 21, 20, 28] and has many applications in networking, and in particular in radio sensor networks. Some of the practical applications include the construction of a backbone for wireless networks, a foundation for routing and for clustering of nodes, and generating spanning trees to reduce communication costs [22, 28].

Many probabilistic algorithms have been suggested in the past three decades to select a MIS in a network, however all either assume nodes know the network topology to a certain extent, and/or that communication between nodes is by messages of logarithmic length in the network size. In contrast, as we have shown, flies solve the MIS problem without knowing the network topology nor sending multi bit messages. Inspired by a detailed study of the fly selection process we derived [1] a $O(\log^2 n)$ rounds synchronous MIS algorithm which does not require any knowledge about the topology except for an upper bound on the total number of nodes and uses only one bit unary messages (just a signal, similar to a beep as in [5]) with an overall optimal message complexity.

While biological systems are not always optimal when it comes to run time complexity, they are usually very robust and adaptable. These systems usually use simple basic building blocks for the communication between cells or molecules. Software systems, on the other hand, are often sensitive to modeling assumptions and can fail when these assumptions do not hold. It is thus of interest to study the basic biological communication model in more detail and to derive algorithms that, even if they do not mimic directly the activity of the biological systems, can accommodate the limited set of assumptions they rely on. In this paper we introduce *the fly* model in which we formally capture our assumptions regrading the a biological communication model. Based on this model we extend our algorithm from [1] in several ways making it applicable to a wider set of applications. First, we introduce a new algorithm for MIS under this model that does not require any knowledge regarding the topology of the network. That is, we assume that nodes do not know how many neighbors they have, cannot count their neighbors and have no information regarding the number of nodes in the entire network. The new algorithm also removes the assumption used by the algorithm in [1] that all nodes wake up simultaneously at the same round, while maintaining its complexities. Next, in Section 5, we extend it to the beeping model [5], where unlike our original model nodes cannot listen while broadcasting a signal. That is, our original algorithm assumes a beeping with collision detection model and we extend it to remove the collision detection assumption, paying a factor of $\log n$ in its round complexity (though still requiring no information about the network topology). Finally, in Section 6 we prove a lower bound of $\Omega(\log^2 n)$ on the number of rounds, under the restriction (which is obeyed by all the algorithms we present and by the algorithm in [1]), that all the nodes broadcast with the same probability in each round (but possibly different from round to round).

2 Related work

While several methods were suggested for solving MIS in the case of symmetric processors, these methods have always assumed that nodes know something about the local or global topology of the network. Most previous methods assumed that nodes know the set of active neighbors each has at each stage of the execution. Moscibroda and Wattenhofer [21] were the first to consider the more realistic setting in which such information is not known. They consider MIS selection in multi-hop networks where only an upper bound is known regarding the number of nodes in the network and with asynchronous wake up. Unlike our model they do not assume that nodes are equipped with collision detectors. However, in addition to the upper bound assumption their model allows for (and their algorithm uses) messages whose size is a function of the number of nodes in the network. In contrast, for the unary message model, collision detectors maybe more realistic [23]. For that model Moscibroda and Wattenhofer present a $(\log^2 n)$ algorithm in the unit disk graph model. In [25] Schneider and Wattenhofer presented a $O(\log^* n)$ algorithm for the more general polynomially bounded growth graphs. However, that algorithm does not assume that collisions occur and so, as the authors write, is less appropriate for wireless networks. In [24] Schneider and Wattenhofer study the power of collision detection in broadcast networks, such as in our model. Their study assumes that n is known and that the processors unique IDs are in $[1, n]$, all represented by the same number of bits. In [24] the asynchronous wake-up problem is also solved, in a way similar to the current paper.

In [20] the importance of the bit complexity in distributed computing algorithm is discussed. They present a $O(\log n)$ algorithm for MIS using only constant size messages for a total of $O(\log n)$ bits per channel. However, unlike our model they do not account for collisions and assume that nodes can count the number of neighbors they have and distinguish between messages sent by different neighbors. In addition, their algorithms uses messages that are larger than 1 bit and while each channel only delivers $O(\log n)$ bits, the overall message complexity may be much larger than $O(n)$ depending on the connectivity of the network. In contrast, our algorithm has an overall linear message complexity.

Several papers discuss reasonable assumptions regarding collision detection and wake-up models in wireless sensor networks. These were recently classified based on the strength of the assumptions involved [12]. It was argued that collisions can be affected even by nodes that are not necessarily in direct contact with the receiving node [23]. However, these models refer to the ability to receive complete packets. If the only requirement is to be able to sense the medium to determine whether a message was sent or not, both the range of each node and its ability to identify collisions is increased. Note that in the fly model, any message that is sent contains the same information. This type of communication was recently termed the beeping model [5] and in that paper the authors presented an algorithm for interval coloring in this model assuming no collision detection. Their algorithm runs in $O(\Delta \log n)$ rounds where Δ is the maximal degree of any node in the network. In contrast, we present an $O(\log^3 n)$ rounds algorithm for MIS in this model for an arbitrary degree graph.

3 Model

We assume a synchronous communication model on an arbitrary graph $G = (V, E)$ where the vertices V represent processors and the edges represent pairs of processors that hear each other. This is the same model as the discrete model in [5], except for collision detection. The model assumed in this paper, called *the fly* model, applies to both, our understanding of how cells interact as well as to radio sensor networks. Specifically, we assume the following:

1. Synchronous rounds - We assume a synchronous network, as the discrete model in [5]. Nodes are *asynchronous* except that the transmission slots are discrete and start at the same time in all the nodes. It is easy to transform the results also to the un-slotted case, where all slots are of the same length but their starting times are not synchronized, the round complexity of the un-slotted version is twice (factor of 2) that of the corresponding slotted version.

2. Asynchronous wake-up: The algorithm is initiated by an arbitrary subset of the nodes that wake up spontaneously at arbitrary rounds, obviously not necessarily all together. Notice, that under this assumption the time complexity of the algorithm is at least the network diameter, the number of rounds it would take a woken up node to wake up all other nodes. However, for our *time complexity* we consider for each node the number of rounds it has been active in the algorithm. We discuss this in more details below. This way of measuring time complexity is different but equivalent to that of [24], where the time complexity of the algorithm is the number of rounds from the time the last processor woke up until the algorithm terminates.

3. No knowledge about the network topology or the number of neighbors: Cells physically interact, however, cells (in biology) cannot tell exactly how many neighbors they have since the geometry of the interaction depends on the cell shape which can change over time. Correspondingly we assume a network in which nodes do not know the number of neighbors each has, nor do they have any other knowledge about the network topology.

4. Beeping communication: In biological system, cells communicate by secreting certain proteins that are sensed ("heard") by neighboring cells [4]. This is similar to a node in a radio network transmitting a carrier signal which is sensed ("heard") by its neighbors. Thus we assume nodes communicate only by transmitting such signals, called beeps[5]. All nodes within range of a beep transmitting station hear the beep if they listen.

5. Collision detection: Cells can distinguish silence from the case that at least one neighbor is transmitting a beep. Depending on the specific process cells can either send and receive beeps at the same round or cannot receive at the time that they transmit a beep, hence can only hear a beep if they are not transmitting at the same time [3]. In the corresponding network model we assume that in each round a node may broadcast a beep-signal or stay silent, and in each round a node hears if any of its neighbors (not including itself) has broadcasted or not. A node hearing a beeping signal cannot tell which of its neighbors beeped, or if one or more have beeped. We distinguish between two variations: whether a node can listen at the same round in which it is beeping or not as in [5].

We define the *time (round) complexity* in our model as follows: consider for each node the active time as the number of rounds it has been active in the algorithm. I.e., from the first round in which the node woke up (spontaneously or by a neighbor) until it and its immediate neighbors exit the algorithm. The time complexity is then the maximum active time (expected) over all the nodes. This way of measuring time complexity is different but equivalent to that of [24], where the time complexity of the algorithm is the number of rounds from the time the last processor woke up until the algorithm terminates. The *message complexity* is the expected total number times nodes broadcast a signal/beep to their neighbors.

This model is appropriate for many distributed computing applications, most notably ad-hoc sensor networks which operate in topologies that are initially unknown, are required to conserve energy and can more accurately detect (beeping) signals when compared to larger messages [23].

4 An iterative algorithm for MIS

Roughly speaking, since in the fly model we have no knowledge about node degrees, the algorithm we presented in [1] worked by sweeping the space of possible node degrees from the high degree nodes (trials with small probabilities) to the lower degree nodes. Given an upper bound n on the number of nodes in the network, all nodes would start by flipping coins with probability $1/n$ for $C \log n$ rounds. If a node was not connected to a member of the MIS by the end of this set of rounds it would double its probability and repeat this process until it, or one of its neighbors, is selected as a MIS member. We have proven that when nodes double their probability from $1/k$ to $2/k$, with very high probability there are no nodes in the network with more than k neighbors.

We have shown that the above algorithm works in $O(\log^2 n)$ rounds and its expected number of messages are optimal (linear in the number of nodes). However, since the algorithm starts with coin flip probabilities proportional to the total number of nodes in the network, it requires knowledge regarding an upper bound on this number. Such knowledge is not always available. Even when a rough estimate about the total number of nodes exists, there could be cases where not all nodes participate (either to conserve energy or because they have other constrains). In such cases the upper bound n can be exponential in the actual number of nodes participating in the algorithm leading to run time that are linear in the number of nodes.

Below we present a new algorithm for solving MIS in the fly model that does not rely on *any* topological knowledge and does not requirer an upper bound on the number of nodes in the network. Unlike our algorithm in [1], in the new algorithm nodes go through several iterations in which they gradually decrease their probability of being selected. The run time of the algorithm is still $O(\log^2 n)$ as we prove below. There is, of course, a price to be paid. In our case the price comes from an increase in message complexity as we discuss below. In addition to eliminating the dependence on any topological information, the new algorithm can also handle asynchronous wakeups improving upon the synchronous wakeup model of [1].

The algorithm is given in Figure 1. The algorithm proceeds in phases each consisting of x steps where x is the total number of phases performed so far (the phase counter). Assuming all nodes start simultaneously at the same round, step i of each phase consists of two exchanges. In the first exchange nodes beep with probability $1/2^i$, and in the second exchange a node that beeped in the first and did not hear a beep from any of its neighbors, beeps again, telling its neighbors it has joined the MIS and they should become inactive and exit the algorithm.

4.1 Asynchronous wakeup

To remove the assumption that all nodes start together nodes that woke up spontaneously propagate a wave of wake-up beep throughout the network. Upon hearing the first beep, which must be the wake up beep, a node broadcasts the wake up beep on the next round, and then waits one round to ensure none of its neighbors is still waking up. This ensures that all neighbors of a node wakeup either at the same round as that node or one round before, or after, that node. Due to these possible differences in wakeup time, we divide each exchange between 3 rounds. Nodes listen in all three rounds to incoming messages. During the second round of the first exchange each active node broadcasts a message to its neighbors with probability p_i (the value of p_i is given in the algorithm). The second exchange also takes three rounds. A node that has broadcasted a message in the first exchange joins the MIS if none of its neighbors had broadcasted in any of the three round of the first exchange. Such node broadcasts again a message in the second round of the second exchange telling its neighbors to terminate the algorithm.

The algorithm is presented in Figure 1.

```
MISNoTopology()
  Wait to hear the first wake up beep
  **If** received a beep (the first)
      **then** broadcast wake up beep to all neighbors   /* Make sure your neighbors wake up too
  Wait one round                              /* while your neighbor(s) wake up their neighbors
  x = 0
  **Repeat**                                                    /*Until exiting the algorithm
    x = x + 1                                                   /* $2^x$ is current size estimate
    **for** $i = 0$ to $x$ **do:**                                              /* $\log 2^x$ phases
          ** **exchange 1** ** **with** 3 **rounds**
      Listen for 1 round                                                            /* round 1
      $v = 0$
      With probability $\frac{1}{2^i}$ broadcast signal and set $v = 1$ ;                     /* round 2
      Listen for 1 round                                                            /* round 3
      **If** heard a signal in any round of exchange 1 **then** $v = 0$
          ** **exchange 2** ** **with** 3 **rounds**
      Listen for 1 round                                                            /* round 1
      **If** $v == 1$ **then** Broadcast signal; Join MIS; and exit                   /* round 2
      Listen for 1 round                                                            /* round 3
      **If** heard a signal in any round of exchange 2 **then** become *inactive*; and exit
      **end**
  **end**
```

Figure 1: Algorithm 1: MIS in the beeping model *with* collision detection.

4.2 Safety properties

While the algorithm in [1] used a different set of coin flip probabilities, it relied on a similar two exchanges structure to guarantee the safety properties of the algorithm in the fly model. In that paper each exchange was only one round (since we assumed synchronous wakeup). We thus need to show that replacing each one round exchange with a three round exchange does not affect the MIS safety properties of our algorithm. We will thus start by proving that the termination lemma from [1], which relies on the fact that all neighbors are using the same probability distribution in each exchange, still holds.

Lemma 1 *All messages received by node j in the first exchange of step i were sent by processors using the same probability as j in that step.*

Proof. Let k be a neighbor of j. If k started at the same round as j (both woke up at the same round) then they are fully synchronized and we are done. If k started before j then the first message k sent has awakened j. Thus, they are only one round apart in terms of execution. Any message sent by k in the 2nd round of the $1st$ exchange of step i would be received by j in the first round of that exchange. Similarly, if k was awakened after j it must have been a 1 round difference and j would receive k's message for the first exchange of step i (if k decided to broadcast) in the third round of that exchange. Thus, all messages received by j are from processors that are also in step i and so all processors from which j receives messages in that exchange are using the same probability distribution. ■

Note that a similar argument would show that all messages received in the second exchange of step i are from processors that are in the second exchange of that step. Since our safety proof only relies on the coherence of the exchange they still hold for this algorithm. Notice also that by adding a listening round at the beginning and end of each exchange the algorithm now works in the un-slotted model (with at most doubling the round complexity).

4.3 Runtime analysis

After establishing the safety guarantees, we next prove that with high probability all nodes terminate the algorithm in $O(\log^2 n)$ time where n is the number of nodes that participate in the algorithm. Let d_v be the number of active neighbors of node v. We start with the following definition [2]:

Definition: v is Good if it has at least $d_v/3$ active neighbors u, s.t., $d_u \leq d_v$.

Lemma 4.4 from [2]: In every graph G at least half of the edges touch a Good vertex. Thus, $\sum_{v \in Good} d_v \geq |E|/2$.

Note that we need less than $O(log^2 n)$ steps to reach $x \geq \log n$ since each phase until $x = \log n$ has less than $\log n$ steps. When $x \geq \log n$, the first $\log n$ steps in each phase are using the probabilities: $1, 1/2, 1/4, ..., 2/n, 1/n$. Below we show that from the time $x = \log n$, we need at most $O(\log n)$ more phases to guarantee that all processors terminate the algorithms with high probability.

Lemma 2 *The expected number of edges deleted in a phase (with more than $\log n$ steps) is $\Omega(|E|)$*

Proof. Fix a phase j, and fix a Good vertex v. We show that the expected number of edges incident with v that are deleted in this phase is $\Omega(d_v)$. Assume that at the beginning of phase j, $2^k \leq d_v \leq 2^{k+1}$ for some $k < \log n$. If when we reach step $i = k$ in phase j at least $d_v/20$ edges incident with v were removed already we are done. Otherwise, at step i there are still at least $d_v/3 - d_v/20 > d_v/4 \geq 2^{k-2}$ neighbors u of v with $d_u \leq d_v$. Node v and all its neighbors u are flipping coins with probability $\frac{1}{2^k}$ at this step and thus the probability that at least one of them would broadcast is:

$$p(v\ or\ u,\ neighbor\ of\ v\ with\ d_u\ \leq d_v,\ broadcasts) \geq 1 - (1 - \frac{1}{2^k})^{2^{k-2}} \cong 1 - 1/e^{1/4}$$

On the other hand, all such nodes u, and v, have less than 2^{k+1} neighbors. Thus, the probability that a node from this set that broadcasts a message does not collide with any other node is:

$$p(\textit{no collisions}) \geq (1 - \frac{1}{2^k})^{2^{k+1}} \cong 1/e^2$$

Thus, in every phase a Good node v has probability of at least $(1 - \frac{1}{e^{1/4}})\frac{1}{e^2} \geq \frac{1}{2^7}$ to be removed. Thus, the probability that v is removed is $\Omega(1)$ which means that the expected number of edges incident with v removed during this phase is $\Omega(d_v)$.
Since half the edges touch a Good node, by the linearity of expectation the expected number of edges removed in each phase is $\geq \frac{1}{2}\Omega(\sum_{v \in Good} d_v) = \Omega(|E|)$.

Note that since the number of edges removed in a phase in a graph (V, E) is clearly always at most $|E|$, the last lemma implies that for any given history, with probability at least $\Omega(1)$, the number of edges removed in a phase is at least a constant fraction of the number of edges that have not been deleted yet. Therefore there are two positive constants p and c, both bounded away from zero, so that the probability that in a phase at least a fraction c of the number of remaining edges are deleted is at least p. Call a phase successful if at least a fraction c of the remaining edges are deleted during the phase.

By the above reasoning, the probability of having at least z successful phases among m phases is at least the probability that a binomial random variable with parameters m and p is at least z. By the standard estimates for Binomial distributions, and by the obvious fact that $O(\log |E|/c) = O(\log n)$, starting from $x = \log n$ we need an additional $O(\log n)$ phases to finish the algorithm. Since each of these additional $O(\log n)$ phases consists of $O(\log n)$ steps, and since as discussed above until $x = \log n$ we have less than $O(\log^2 n)$ steps, the total running time of the algorithm is $O(\log^2 n)$.

■

4.4 message complexity

Note that unlike our algorithm from [1], the message complexity of this algorithm is not optimal. In fact, a lower bound for the message complexity is $\Omega(n)$ even if the size of the MIS is $O(1)$. To see this consider the complete graph with n processors. Since only 1 processor can be in the MIS for this graph we need a step in which one processor flips a 1 while the rest obtain a 0. Since all processors send a message in step 0 of phase 1, the expected number of send events is at least n.

Note that this is tight, up to a logarithmic factor, for any graph on n vertices. Indeed, the total number of phases is $O(\log n)$ and the expected number of broadcast messages a node sends during a phase is bounded by $\sum_i \frac{1}{2^i} < 2$, providing an $O(n \log n)$ upper bound.

5 Beeping model with no collision detection

In the previous section we assumed a collision detection model that allowed nodes to listen to incoming messages in the same round they are sending. As noted in Section 3, the beeping model [5] as well as other communication models [12] are more strict and only allow listening in rounds in which a processor does not broadcast. To extend our algorithm to this model we increase the number of exchanges from 3 to x. Prior to starting the first exchange each *active* processor flips a coin with the same probability as in Algorithm 1. If the flip outcome is 0 (tail) the processor only listens in the next cx exchanges (for a constant c discussed below). If the flip outcome is 1 the processor sets $v = 1$ and selects at random (with equal probability) half of the following cx exchanges. During the selected exchanges the processor broadcasts a beep. In all other exchanges it listens. If at any of the exchanges it listens it hears a beep it sets $v = 0$ and stops broadcasting (even in the selected exchanges). If a node hears a beep during these exchanges it does not exit the algorithm. Instead, it denotes the fact that one of its neighbors beeped and sets itself to be inactive. If it does not hear a beep in any of the exchanges of a future phase it becomes active and continues as described above. Similarly, a node that beeped and did not hear any beep in a specific step (indicating that it can join the MIS) continues to beep indefinitely (by selecting half the exchanges in all future steps and following the algorithm above).

The guarantees for this model differ from the guarantees of the stronger collision detection model. Specifically, the algorithm guarantees that after a certain time (which depends on the network size and is derived below), all MIS members are fixed, and the safety requirements hold (all nodes not in the MIS are connected to a node in the MIS and no two MIS members are connected). Until this time nodes can decide to become MIS members and later drop from the set if they find out that one of their neighbors has also decided to join the MIS. Since nodes do not have an estimate of the network size the processors continue to perform the algorithm indefinitely. If an upper bound on the network size is available we can derive a specific stopping criteria for this model as we discuss below.

The algorithm is presented in Figure 2.

The main difference between this algorithm and Algorithm 1 a set of competition exchanges that are added at the end of each coin flip. The number of competition exchanges is equal to the current phase counter (which serves as the current estimate of the network size). Initially the competition rounds are short and so they would not necessarily remove all collisions. We require that nodes that attempt to join continue to participate in all future competition rounds (when $v = 1$). Processors that detect a MIS member as a neighbor set z to 1 and do not broadcast until they go through one complete set of competition exchanges in which they do not hear any beep. If and when this happens they set $z = 0$ and become potential MIS candidate again.

While not all collisions will be resolved at the early phases, when $x \geq \log n$ each set of competition exchanges is very likely to remove all collisions. We prove below that once we arrive at such x values, all collisions are resolved with very high probability such that only one processor in a set of colliding processors remains with $v = 1$ at the end of these competition exchanges. From there, it takes another $O(\log n)$ phases to select all members of the MIS as we have shown for Algorithm 1. Since each such phase takes $(O(\log n)$ steps with each step taking $O(\log n)$ rounds for the competition, the total run time of the algorithm is $O(\log^3 n)$.

Below we prove that if two neighbors in the network have $v = 1$ after the coin flip in step i in a phase with $x \geq \log n$, then with high probability one would set $v = 0$ at the end of step i of that phase and so at most one of them enters the MIS.

Lemma 3 *Assume processor y collided with one or more of its neighbors setting $v = 1$ in step i of phase $x \geq \log n$. Then the probability that y would still be colliding with any of its neighbors at the end of the cx competition exchanges for step i is $\leq \frac{1}{n^{c/3}}$.*

Proof. If at any of the exchanges in this step all neighbors of y have $v = 0$ we are done. Otherwise in each exchange, with probability at least 1/4 y decided not to broadcast whereas one of its colliding neighbors decided to broadcast. Thus, the probability that y does not resolve its collision in a specific exchange is $\leq 3/4$. Since there are $(c \log n)$ exchanges in this step, the probability that y is still colliding at the end of these competition exchanges $\leq (\frac{3}{4})^{c \log n} \leq \frac{1}{n^{c/3}}$. ■

Note that if a collision is not resolved in a specific exchange the colliding nodes continue to broadcast in the following phase. As we proved in the previous section, if all collisions are resolved

```
Wait to hear the first wake up beep
**If** received a beep (the first)
    **then** broadcast wake up beep to all neighbors    /* Make sure your neighbors wake up too
Wait one round                                        /* while your neighbor(s) wake up their neighbors
$x = 0$;$v = 0$; $z = 0$
**Repeat forever**
  x = x + 1
  **for** $i = 0$ to $x$ **do:**
    if $v = 0$ & $z = 0$
      With probability $\frac{1}{2^i}$ set $v = 1$
    endif
    Set each entry in a vector $X$ of size $cx$ to 1 with probability 1/2, the rest are 0 /* $c$ is a constant
    $z = 0$
         ** $cx$ **competition exchanges**
    For $k = 1$ to $cx$
      Listen for 1 round. If heard a signal set $v = 0$, $z = 1$ /* $z = 1$ means connected to node in MIS
      If $v = 0 \parallel X(k) = 0$
        Listen for 1 round. If heard a signal set $v = 0$, $z = 1$
      Else                                                  /* $v = 1$ & $X(k) = 1$
        Beep one round
      EndIf
      Listen for 1 round. If heard a signal set $v = 0$, $z = 1$
    End For (k)
  End For (j)
**end**
```

Figure 2: Algorithm 2: MIS in the beeping model *without* collision detection.

in the $O(\log n)$ phases that follow the phase $x = \log n$ the algorithm will result in a MIS set with very high probability. Since we only need $O(\log^2 n) < n$ steps for this, and we have n nodes, the probability that there exists a step and a node in phase $x \geq \log n$ such that a node that collided during this step with a neighbor does not resolve this collision in that step is smaller than $\frac{1}{n^{c/3-2}}$. Thus, with probability $\geq 1 - \frac{1}{n^{c/3-2}}$ all collisions are detected and the MIS safety condition holds.

5.1 Stopping criteria when an upper bound on network size exists

The above algorithm leads to a MIS set and does not require knowledge of the network size. However, the time it takes to reach the MIS set is a function of the size of the network and so if node do not have an estimate of this number they cannot terminate the algorithm and need to indefinitely listen to incoming messages. Note that, as the analysis above indicates, if a rough estimate on the network size n exists, nodes can terminate the algorithm when $x = 2\log n + 1$. At that phase we have a high probability that every collision that has occurred during the last $\log n$ phases has been resolved ($\geq 1 - \frac{1}{n^{c/3-2}}$) and as proved in the previous section when all collisions are resolved the algorithm terminates with very high probability.

6 A Round Complexity Lower bound

The algorithms we have presented so far shared a common theme: At each exchange all processors broadcast with the same probability. Note that this is not a requirement of the model and nodes can change their coin flip probability based on the history of messages they received which may lead to different probabilities for different nodes. However, for the lower bound proof we focus on such algorithms. Specifically, we assume synchronous rounds where at each exchange all processors flip coins with the same probability p_t (though p_t can be controlled by an adversary and set just before the exchange begins). We allow a round of coin flips that can be followed by several rounds of deterministic behavior and, combined, they consist of a single exchange in the algorithm. Our lower bound proof is based on the following graph construction. We consider a graph consisting of a disjoint union of complete bipartite graphs. For each i between 1 and $(\log n)/4$ the graph has at least $n^{0.7}$ pairwise disjoint complete bipartite graphs of type i, that is, $n^{0.7}$ bipartite graphs with 2^i vertices in each side.

We first define the notion of a failed broadcast for a complete bipartite graph. We say that a broadcast failure has occurred in round t for such graph if

1. No vertex in the graph broadcasts, or,
2. On both sides of the graph at least one vertex broadcasts a message.

Lemma 4 *Let G be a complete bipartite graph. If all rounds up to t resulted in broadcast failure in G then no vertex in G has joined the MIS.*

In Appendix A we provide the proof of the lemma 4 and we complete the proof that $\Omega(\log^2 n)$ phases are required for any algorithm that solves the MIS problem in this model.

7 Conclusions and future work

We have shown that using a severely constrained communication model based on a biological process, we can derive efficient algorithms for the MIS selection problem. Our algorithm can work with two different types of collision detection models. We have also presented a lower bound for a variant of the fly model in which coin flipping probability is constrained so that all processors flip coins with the same probability in each round. Algorithm 1 we presented for the MIS was derived based on insights from the SOP selection process in flies which solves a variant of the MIS selection problem [26]. Other biological systems, including various types of networks in the cell, are also addressing distributed computing problems. These include resource allocation during cell division [19], routing [27] and backup and fault tolerance in regulatory networks [8]. It is our hope that motivating ideas for improving other distributed computing tasks can be derived from studies of these biological processes.

References

[1] Y. Afek, N. Alon, O. Barad, E. Hornstein, N. Barkai, Z. Bar-Joseph, "A Biological Solution to a Fundamental Distributed Computing Problem" Science, vol. 331, Jan. 2011, pp. 183-185. (See http://www.sciencemag.org/content/suppl/2011/01/10/331.6014.183.DC1/Afek-SOM.pdf for the supplement materials.)

[2] N. Alon, L. Babai, and A. Itai, "A fast and simple randomized parallel algorithm for the maximal independent set problem," Journal of Algorithms, vol. 7, 1986, pp. 567-583.

[3] O. Barad, D. Rosin, E. Hornstein, and N. Barkai, "Error minimization in lateral inhibition circuits," Science Signaling, vol. 3, Jul. 2010, p. ra51.

[4] J.R. Collier, N.A. Monk, P.K. Maini, and J.H. Lewis, "Pattern formation by lateral inhibition with feedback: a mathematical model of delta-notch intercellular signalling," Journal of Theoretical Biology, vol. 183, Dec. 1996, pp. 429-446.

[5] A. Cornejo and F. Kuhn, "Deploying Wireless Networks with Beeps," Distributed Computing, 24th International Symposium, DISC 2010, 2010, pp. 148-162.

[6] J. Ernst, O. Vainas, C.T. Harbison, I. Simon, and Z. Bar-Joseph, "Reconstructing dynamic regulatory maps," Molecular Systems Biology, vol. 3, 2007, p. 74.

[7] G. Frederickson, and N. Lynch, "Electing a leader in a synchronous ring", JACM, vol. 34(1), January 1987, 98-115.

[8] A. Gitter, Z. Siegfried, M. Klutstein, O. Fornes, B. Oliva, I. Simon, and Z. Bar-Joseph, "Backup in gene regulatory networks explains differences between binding and knockout results," Molecular Systems Biology, vol. 5, 2009, p. 276.

[9] L. Glass, "Synchronization and rhythmic processes in physiology," Nature, vol. 410, Mar. 2001, pp. 277-284.

[10] K.J. Ishii, S. Koyama, A. Nakagawa, C. Coban, and S. Akira, "Host innate immune receptors and beyond: making sense of microbial infections," Cell Host & Microbe, vol. 3, Jun. 2008, pp. 352-363.

[11] R. Kafri, A. Bar-Even, and Y. Pilpel, "Transcription control reprogramming in genetic backup circuits," Nature Genetics, vol. 37, Mar. 2005, pp. 295-299.

[12] F. Kuhn, N. Lynch, K. Newport, R. Oshman, and A. Richa, "Broadcasting in unreliable radio networks.," Proceeding of the 29th ACM SIGACT-SIGOPS symposium on Principles of distributed computing (PODC '10), 2010, pp. 336-345.

[13] F. Kuhn, T. Moscibroda and R. Wattenhofer, "On the locality of bounded growth graphs" PODC 2005, 60-68.

[14] F. Kuhn, T. Moscibroda, T. Nieberg, and R. Wattenhofer, "Fast Deterministic Distributed Maximal Independent Set Computation on Growth-Bounded Graphs" DISC 2005: 273-287.

[15] F. Kuhn, T. Moscibroda and R. Wattenhofer "The price of being near-sighted". SODA 2006: 980-989

[16] F. Kuhn, T. Moscibroda and R. Wattenhofer: "What cannot be computed locally!" PODC 2004: 300-309

[17] M. Luby, "A simple parallel algorithm for the maximal independent set problem," Proceedings of the seventeenth annual ACM symposium on Theory of computing - STOC 85, Providence, Rhode Island, United States: 1985, pp. 1-10.

[18] N. Lynch, Distributed algorithms, San Francisco Calif.: Morgan Kaufmann Publishers, 1996.

[19] M. Méchali, "Eukaryotic DNA replication origins: many choices for appropriate answers," Nature Reviews. Molecular Cell Biology, vol. 11, Oct. 2010, pp. 728-738.

[20] Y. Métivier, J. Robson, N. Saheb-Djahromi, and A. Zemmari, "An optimal bit complexity randomized distributed MIS algorithm," Distributed Computing, 2010, pp. 1-10.

[21] T. Moscibroda and R. Wattenhofer, "Maximal independent sets in radio networks," Proceedings of the twenty-fourth annual ACM SIGACT-SIGOPS symposium on Principles of distributed computing - PODC '05, Las Vegas, NV, USA: 2005, p. 148.

[22] D. Peleg, Distributed computing : a locality-sensitive approach, Philadelphia: Society for Industrial and Applied Mathematics, 2000.

[23] C. Scheideler, A. Richa, and P. Santi, “An $O(\log n)$ dominating set protocol for wireless ad-hoc networks under the physical interference model,” Proceedings of the 9th ACM international symposium on Mobile ad hoc networking and computing (MobiHoc ’08), 2008, pp. 91-100.

[24] J. Schneider and R. Wattenhofer, “What Is the Use of Collision Detection (in Wireless Networks)?,” Proceedings of the 24th International Symposium, (DISC)2010, Cambridge, MA, USA, September 13-15, 2010. pp. 133-147.

[25] J. Schneider and R. Wattenhofer, “A $\log^*$ distributed maximal independent set algorithm for growth-bounded graphs.,” Proceedings of the twenty-seventh ACM symposium on Principles of distributed computing (PODC ’08), 2008, pp. 35-44.

[26] P. Simpson, “Notch signalling in development: on equivalence groups and asymmetric developmental potential,” Current Opinion in Genetics & Development, vol. 7, Aug. 1997, pp. 537-542.

[27] A. Tero, S. Takagi, T. Saigusa, K. Ito, D.P. Bebber, M.D. Fricker, K. Yumiki, R. Kobayashi, and T. Nakagaki, “Rules for biologically inspired adaptive network design,” Science (New York, N.Y.), vol. 327, Jan. 2010, pp. 439-442.

[28] P. Wan, K.M. Alzoubi, and O. Frieder, “Distributed Construction of Connected Dominating Set in Wireless Ad Hoc Networks,” Mobile Networks and Applications, vol. 9, 2004, pp. 141-149.

A A Round Complexity Lower bound

Proof. of Lemma 4. Next we show that if a failure occurs up to and including round t than there is a non zero probability that any node j in G has a completely symmetric neighbor k in G at round t. Thus, if j joins the MIS there is a non zero probability that k joins the MIS as well, in the same round, which violates the requirements for a MIS. We prove this by induction on t. *Base:* For $t = 0$ this is trivial and stems from our initial assumptions. For $t = 1$, if the failure is of type 1 all processors are clearly symmetric (both did not send or receive any message from the time the algorithm started). If the failure is of type 2 we distinguish between two cases. If j has broadcasted in phase 1 let k be a neighbor of j that has also broadcasted in that round. Both j and k send and receive a message and so they are symmetric. If j did not broadcast at phase 1 than $p_t < 1$ and there is a positive probability that there exists a node k that is a neighbor of j and did not broadcast at phase 1.

Induction step: We assume correctness for $t - 1$ and prove for t. If the failure at round t is of type 1 then all nodes that were symmetric in phase $t - 1$ remain symmetric in phase t. If the failure is of type 2 then let j and k be two symmetric neighbors at the beginning of phase t. Since both received messages and both use the same probability for broadcasting an argument similar to the one used above shows that there is a positive probability that they would remain symmetric at the end of this phase as well. ■

We now compute the probability that broadcast failure occurs for a specific type of a bipartite graph. Let p_t^i be the probability of such broadcast failure for one complete bipartite graph G with 2^i vertices on each side at phase t. Assume the broadcast probability for this phase is $p = p(t)$. Then:

$$p_t^i = (1-p)^{2^{i+1}} + \left(1 - (1-p)^{2^i}\right)^2$$

The left hand side of this sum accounts for failure of type 1 (no node broadcasts) while the right hand side accounts for failures of type 2.

If $p = 0$ then we are guaranteed that a broadcast failure of type 1 occurs in G. Otherwise let k be an integer such that $\frac{1}{2^{k+1}} < p \leq \frac{1}{2^k}$. For $j = 0, \ldots, (k-1)$ we use the left side of the above sum to obtain a lower bound on the broadcast failure probability:

$$(1-p)^{2^{(j+1)}} \geq \left(1 - \frac{1}{2^k}\right)^{2^{j+1}} \cong \left(\frac{1}{e}\right)^{\frac{1}{2^{k-j-1}}}$$

For $j = k+1, \ldots, (\log n)/4$ we use the right hand side of the sum to bound the broadcast failure probability:

$$\left(1 - (1-p)^{2^j}\right)^2 = 1 - 2(1-p)^{2^j} + (1-p)^{2^{j+1}} \geq 1 - 2(1-p)^{\frac{p}{p}2^j} = 1 - 2\left((1-p)^{1/p}\right)^{p2^j}$$

$$= 1 - 2e^{-p2^j} \geq 1 - 2e^{-2^j/2^k} = 1 - 2e^{-2^{j-k}}$$

Finally, for $j = k$ we use the left hand side of the sum again to get:

$$(1-p)^{2^{j+1}} \geq \left(1 - \frac{1}{2^k}\right)^{2^{k+1}} \cong \frac{1}{e^2} > 0.1$$

We now compute the product of p_t^j over all possible values of j $(0 \leq j \leq \log n/4)$ for a fixed phase t (again we set $p = p(t)$). This product is:

$$\prod_{j=0}^{\log n/4} p_t^j \geq \prod_{j=0}^{\log n/4} \left((1-p)^{2^{(j+1)}} + \left(1 - (1-p)^{2^j}\right)^2\right)$$

$$\geq \left(\prod_{j=0}^{k-1} (1/e)^{\frac{1}{2^{k-j-1}}}\right) 0.1 \left(\prod_{j=k+1}^{(\log n)/4} 1 - 2e^{-2^{j-k}}\right)$$

The left part of this product evaluates to:

$$\left(\prod_{j=0}^{k-1} (1/e)^{\frac{1}{2^{k-j-1}}}\right) = \left(\frac{1}{e}\left(\frac{1}{e}\right)^{1/2}\left(\frac{1}{e}\right)^{1/4} \cdots \left(\frac{1}{e}\right)^{\frac{1}{2^{k-1}}}\right) \geq \frac{1}{e^2}$$

The right part is:

$$\left(\prod_{j=k+1}^{(\log n)/4} 1 - 2e^{-2^{j-k}}\right) \geq 1 - 2\left(\prod_{j=k+1}^{(\log n)/4} e^{-2^{j-k}}\right) \geq 1 - 2\left(\prod_{j=k+1}^{(\log n)/4} e^{-2(j-k)}\right)$$

$$\geq 1 - 2\left(\frac{1}{e^2(1 - e^{-2})}\right) > 0.1$$

The first inequality stems from the fact that $(1-a)(1-b) \geq (1-a-b)$. Thus, for the full product we have:

$$\prod_{j=0}^{\log n/4} p_t^j \geq \prod_{j=0}^{\log n/4} \left((1-p)^{2^{(j+1)}} + \left(1-(1-p)^{2^j}\right)^2 \right) \geq \frac{1}{e^2} * 0.1 * 0.1 \geq .001 \geq \frac{1}{2^{10}}$$

Thus, if we have T phases, no matter what p_t is in each phase, the product over p_t^j for all values of j and all phases $1 \leq t \leq T$ is:

$$\prod_{t=1}^{T} \prod_{j=0}^{(\log n)/4} p_t^j \geq \frac{1}{2^{10T}}$$

By changing order in this product we get:

$$\prod_{t=1}^{T} \prod_{j=0}^{(\log n)/4} p_t^j = \prod_{j=0}^{(\log n)/4} \prod_{t=1}^{T} p_t^j = \prod_{j=0}^{(\log n)/4} \left(\prod_{t=1}^{T} p_t^j \right) \geq \frac{1}{2^{10T}} = \prod_{j=0}^{(\log n)/4} \left(\frac{1}{2^{10T}} \right)^{4/(\log n)}$$

Thus, for at least one value of j we have

$$\prod_{t=1}^{T} p_t^j \geq \left(\frac{1}{2^{10T}} \right)^{4/(\log n)}$$

Set $T = 0.01 \log^2 n$. For such j and T we have:

$$\prod_{t=1}^{T} p_t^j \geq \left(\frac{1}{2^{10T}} \right)^{4/(\log n)} \geq \frac{1}{\sqrt{n}}$$

The above inequality shows that there exists a j such that the probability that a single bipartite graph of type j experiences a broadcast failure in all rounds is at least $\frac{1}{\sqrt{n}}$. Hence the probability it does not experience such failure is at most $(1 - \frac{1}{\sqrt{n}})$. As there are at least $n^{0.7}$ graphs of this type, the probability that none of them fails in all rounds is at most $(1 - \frac{1}{\sqrt{n}})^{n^{0.7}} \cong (1/e)^{n^{0.2}}$. This shows that almost surely there is a bipartite graph that will experience a broadcast failure in all T phases.

Thus we need $\Omega(\log^2 n)$ phases for any algorithm solving the MIS problem under this model.

Comment: In the discussion above we assumed that only one probability value is used by all nodes in each coin flipping round. However, this assumption can be relaxed to allow c different probabilities to be used in each round (for some constant c). For such a model we allow processors to select among a predefined set of c probabilities for *each* phase (they can be different between the phases) based on the history of the messages they received. Note that all nodes in a complete bipartite graph that experienced a broadcast failure up to phase t will use the same probability at that phase since all nodes in such graphs see the same set of incoming messages up to phase t. To show that our proof still holds note that replicating each graph c times when computing the broadcast failure probability leads to a failure probability of $\frac{1}{2^{10c}}$ for each set. We then use $T = (1/100c)(\log^2 n)$ for the reminder of the proof.