

Dynamic Networks of Finite State Machines

Yuval Emek¹ and Jara Uitto²(✉)

¹ Technion, Haifa, Israel

² Comerge AG, Zürich, Switzerland
`jara.uitto@comerge.net`

Abstract. Like distributed systems, biological multicellular processes are subject to dynamic changes and a biological system will not pass the survival-of-the-fittest test unless it exhibits certain features that enable fast recovery from these changes. In most cases, the types of dynamic changes a biological process may experience and its desired recovery features differ from those traditionally studied in the distributed computing literature. In particular, a question seldomly asked in the context of distributed digital systems and that is crucial in the context of biological cellular networks, is whether the system can keep the changing components *confined* so that only nodes in their vicinity may be affected by the changes, but nodes sufficiently far away from any changing component remain unaffected.

Based on this notion of confinement, we propose a new metric for measuring the dynamic changes recovery performance in distributed network algorithms operating under the *Stone Age* model (Emek and Wattenhofer, PODC 2013), where the class of dynamic topology changes we consider includes inserting/deleting an edge, deleting a node together with its incident edges, and inserting a new isolated node. Our main technical contribution is a distributed algorithm for maximal independent set (MIS) in synchronous networks subject to these topology changes that performs well in terms of the aforementioned new metric. Specifically, our algorithm guarantees that nodes which do not experience a topology change in their immediate vicinity are not affected and that all surviving nodes (including the affected ones) perform $\mathcal{O}((C + 1) \log^2 n)$ computationally-meaningful steps, where C is the number of topology changes; in other words, each surviving node performs $\mathcal{O}(\log^2 n)$ steps when amortized over the number of topology changes. This is accompanied by a simple example demonstrating that the linear dependency on C cannot be avoided.

1 Introduction

The biological form of close-range (juxtacrine) message passing relies on designated messenger molecules that bind to crossmembrane receptors in neighboring cells; this binding action triggers a signaling cascade that eventually affects gene expression, thus modifying the neighboring cells' states. This mechanism should feel familiar to members of the distributed computing community as it resembles the message passing schemes of distributed digital systems. In contrast to

nodes in distributed digital systems, however, biological cells are not believed to be Turing complete, rather each biological cell is pretty limited in computation as well as communication. In attempt to cope with these differences, Emek and Wattenhofer [13] introduced the *Stone Age* model of distributed computing (a.k.a. networked finite state machines), where each node in the network is a very weak computational unit with limited communication capabilities and showed that several fundamental distributed computing problems can be solved efficiently under this model.

An important topic left outside the scope of [13] is that of *dynamic topology changes*. Just like distributed digital systems, biological systems may experience local changes and the ability of the system to recover from these changes is crucial to its survival. However, the desired recovery features in biological cellular networks typically differ from those traditionally studied in the distributed computing literature. In particular, a major issue in the context of biological cellular networks, that is rarely addressed in the study of distributed digital systems, is that of *confining* the topology changes: while nodes in the immediate vicinity of a topology change are doomed to be affected by it (hopefully, to a bounded extent), isolating the nodes sufficiently far away from any topology change so that their operation remains unaffected, is often critical. Indeed, a biological multicellular system with limited energy resources cannot afford every cell division (or death) to have far reaching effects on the cellular network.

In this paper, we make a step towards bringing the models from computer science closer to biology by extending the Stone Age model to accommodate four types of dynamic topology changes: (1) deleting an existing edge; (2) inserting a new edge; (3) deleting an existing node together with its incident edges; and (4) inserting a new isolated node. We also introduce a new method for measuring the performance of a network in recovering from these types of topology changes that takes into account the aforementioned confinement property. This new method measures the number of “computationally-meaningful” steps made by the individual nodes, which are essentially all steps in which the node participates (in the weakest possible sense) in the global computational process. An algorithm is said to be *effectively confining* if (i) the runtime of the nodes that are not adjacent to any topology change is $\log^{\mathcal{O}(1)} n$; and (ii) the global runtime (including all surviving nodes) is $(C + 1) \log^{\mathcal{O}(1)} n$, where C is the number of topology changes throughout the execution. In other words, the global runtime is $\log^{\mathcal{O}(1)} n$ when amortized over the number of changes.

Following that, we turn our attention to the extensively studied *maximal independent set (MIS)* problem and design a randomized effectively confining algorithm for it under the Stone Age model extended to dynamic topology changes. This is achieved by carefully augmenting the MIS algorithm introduced in [13] with new components, tailored to ensure fast recovery from topology changes. Being a first step in the study of recovery from dynamic changes under the Stone Age model, our algorithm assumes a *synchronous* environment and it remains an open question whether this assumption can be lifted. Nevertheless, this assumption is justified by the findings of Fisher et al. [14] that model

cellular networks as being subject to a *bounded asynchrony* scheduler, which is equivalent to a synchronous environment from an algorithmic perspective.

Paper’s Organization. An extension of the Stone Age model of [13] to dynamic topology changes is presented in Sect. 2 together with our new method for evaluating the recovery performance of distributed algorithms. In Sect. 3, we describe the details of our MIS algorithm. Then, in Sect. 4.1, we show that each node not affected by a topology change will reach an output state in $\mathcal{O}(\log^2 n)$ rounds. In Sects. 4.2 and 4.3, we finish the analysis of our MIS algorithm by establishing an $\mathcal{O}((C + 1) \log^2 n)$ upper bound on the global runtime. The runtime of the new MIS algorithm is shown to be near-optimal in Sect. 5 by proving that the global runtime of any algorithm is $\Omega(C)$. In the full version of the paper, we show that the runtime of any node u can be further bounded by $\mathcal{O}((C_u + 1) \log^2 n)$, where C_u is the number of topology changes that occur within $\mathcal{O}(\log n)$ hops from u .

Related work. The standard model for a network of communicating devices is the message passing model [28, 35]. There are several variants of this model, where the power of the network has been weakened. Perhaps the best-known variant of the message passing model is the congest model, where the message size is limited to size logarithmic in the size of the input graph [35]. A step to weaken the model further is to consider interference of messages, i.e., a node only hears a message if it receives a single message per round — cf. the *radio network model* [9]. In the *beeping model* [11, 15], the communication capabilities are reduced further by only allowing to send beeps that do not carry information, where a listening node cannot distinguish between a single beep and multiple beeps transmitted by its neighbors.

The models mentioned above focus on limiting the communication but not the computation, i.e., the nodes are assumed to be strong enough to perform unlimited (local) Turing computations in each round. Networks of nodes weaker than Turing machines have been extensively studied in the context of *cellular automata* [16, 33, 41]. While the cellular automata model typically considers the deployment of finite state machines in highly regular graph structures such as the grid, the question of modeling cellular automata in arbitrary graphs was tackled by Marr and Hütt in [31], where a node changes its binary state according to the densities of its neighboring states. Another extensively studied model for distributed computing in a network of finite state machines is *population protocols* [4] (see also [5, 32]), where the nodes communicate through a series of pairwise rendezvous. Refer to [13] for a comprehensive account of the similarities and differences between the Stone Age model and the models of cellular automata and population protocols.

Distributed computing in dynamic networks has been extensively studied [6, 19, 23, 26, 40]. A classic result by Awerbuch and Sipser states that under the message passing model, any algorithm designed to run in a static network can be transformed into an algorithm that runs in a dynamic network with only a constant multiplicative runtime overhead [7]. However, the transformation of Awerbuch and Sipser requires storing the whole execution history and sending

it around the network, which is not possible under the Stone Age model. Some dynamic network papers rely on the assumption that the topology changes are spaced in time so that the system has an opportunity to recover before another change occurs [20, 21, 30]. The current paper does not make this assumption.

The *maximal independent set (MIS)* problem has a long history in the context of distributed algorithms [3, 10, 28, 29, 34, 39]. Arguably the most significant breakthrough in the study of message passing MIS algorithms was the $\mathcal{O}(\log n)$ algorithm of Luby [29] (developed independently also by Alon et al. [3]). Later, Barenboim et al. [8] showed an upper bound of $2^{\mathcal{O}(\sqrt{\log \log n})}$ in the case of polylogarithmic maximum degree and an $\mathcal{O}(\log \Delta + \sqrt{\log n})$ bound for general graphs. For growth bounded graphs, it was shown by Schneider et al. [36] that MIS can be computed in $\mathcal{O}(\log^* n)$ time. In a recent work, Ghaffari studied the *local* complexity of computing an MIS, where the time complexity is measured from the perspective of a single node, instead of the whole network [17]. In a similar spirit, we provide, in addition to a global runtime bound, a runtime analysis from the perspective of a single node in the full version of the paper.

In the radio networks realm, with f channels, an MIS can be computed in $\Theta(\log^2 n/f) + \tilde{\mathcal{O}}(\log n)$ time [12], where $\tilde{\mathcal{O}}$ hides factors polynomial in $\log \log n$. The MIS problem was extensively studied also under the beeping model [1, 2, 37]. Afek et al. [1] proved that if the nodes are provided with an upper bound on n or if they are given a common sense of time, then the MIS problem can be solved in $\mathcal{O}(\log^{\mathcal{O}(1)} n)$ time. This was improved to $\mathcal{O}(\log n)$ by Scott et al. [37] assuming that the nodes can detect sender collision.

On the negative side, the seminal work of Linial [28] provides a runtime lower bound of $\Omega(\log^* n)$ for computing an MIS in an n -node ring under the message passing model [28]. Kuhn et al. [22] established a stronger lower bound for general graphs stating that it takes $\Omega\left(\sqrt{\frac{\log n}{\log \log n}}\right) + \Omega\left(\frac{\log \Delta}{\log \log \Delta}\right)$ rounds to compute an MIS, where Δ is the maximum degree of the input graph. For uniform algorithms in radio networks (and therefore, also for the beeping model) with asynchronous wake up schedule, there exists a lower bound of $\Omega(\sqrt{n/\log n})$ communication rounds [1].

Containing faults within a small radius of the faulty node has been studied in the context of *self-stabilization* [18]. An elegant MIS algorithm was developed under the assumption that the activation times of the nodes are controlled by a *central daemon* who activates the nodes one at a time [27, 38]. In contrast, we follow the common assumption that all nodes are activated simultaneously in each round. In the self-stabilization realm, the performance of an algorithm is typically measured as a function of some network parameter, such as the size of the network or the maximum degree, whereas in the current paper, the performance depends also on the number of failures.

With respect to the performance evaluation, perhaps the works closest to ours are by Kutten and Peleg [24, 25], where the concepts of *mending algorithms* and *tight fault locality* are introduced. The idea behind a fault local mending algorithm is to be able to recover a legal state of the network after a fault occurs, measuring the performance in terms of the number of faults. The term tight

fault locality reflects the property that an algorithm running in time $\mathcal{O}(T(n))$ without faults is able to mend the network in time $\mathcal{O}(T(F))$, where F denotes the number of faults. The algorithm of Kutten and Peleg recovers an MIS in time $\mathcal{O}(\log F)$, but they use techniques that require nodes to count beyond constant numbers, which is not possible in the Stone Age model. Furthermore, they consider transient faults, whereas we consider permanent changes in the network topology.

2 Model

Consider some network represented by an undirected graph $G = (V, E)$, where the nodes in V correspond to the network's computational units and the edges represent bidirectional communication channels. Adopting the (static) Stone Age model of [13], each node $v \in V$ runs an algorithm captured by the 8-tuple $\Pi = \langle Q, q_0, q_{yes}, q_{no}, \Sigma, \sigma_0, b, \delta \rangle$, where Q is a fixed set of *states*; $q_0 \in Q$ is the *initial state* in which all nodes reside when they wake up for the first time; q_{yes} and q_{no} are the *output states*, where the former (resp., latter) represents membership (resp., non-membership) in the output MIS; Σ is a fixed *communication alphabet*; $\sigma_0 \in \Sigma$ is the *initial letter*; $b \in \mathbb{Z}_{>0}$ is a *bounding parameter*; and $\delta : Q \times \{0, 1, \dots, b\}^{|\Sigma|} \rightarrow 2^{Q \times (\Sigma \cup \{\varepsilon\})}$ is the *transition function*. For convenience, we sometimes denote a state transition from q to q' by $(q \rightarrow q')$ and omit the rules associated with this transition from the notation.

Node v communicates with its neighbors by transmitting messages that consist of a single letter $\sigma \in \Sigma$ such that the same letter σ is sent to all neighbors. It is assumed that v holds a port $\phi_u(v)$ for each neighbor u of v in which the last message (a letter in Σ) received from u is stored. Transmitting the designated empty symbol ε corresponds to the case where u does not transmit any message. In other words, when node u transmits the ε letter, the letters in ports $\phi_u(v)$, for all neighbors v of u , remain unchanged. In the beginning of the execution, all ports contain the initial letter σ_0 .

The execution of the algorithm proceeds in discrete synchronous rounds indexed by the positive integers. In each round $r \in \mathbb{Z}_{>0}$, node v is in some state $q \in Q$. Let $\sharp(\sigma)$ be the number of appearances of the letter $\sigma \in \Sigma$ in v 's ports in round r and let $\langle \min\{\sharp(\sigma), b\} \rangle_{\sigma \in \Sigma}$ be a Σ -indexed vector whose σ -entry is set to the minimum between $\sharp(\sigma)$ and the bounding parameter b . Then the state q' in which v resides in round $r + 1$ and the message σ' that v sends in round r (appears in the corresponding ports of v 's neighbors in round $r + 1$ unless $\sigma' = \varepsilon$) are chosen uniformly at random among the pairs in

$$\delta(q, \langle \min\{\sharp(\sigma), b\} \rangle_{\sigma \in \Sigma}) \subseteq Q \times (\Sigma \cup \{\varepsilon\}).$$

This means that v tosses an unbiased die (with $|\delta(q, \langle \min\{\sharp(\sigma), b\} \rangle_{\sigma \in \Sigma})|$ faces) when deciding on q' and σ' .

To ensure well-defined state transitions, we require that $|\delta(q, x)| \geq 1$ for all $q \in Q$ and $x \in \{0, 1, \dots, b\}^{|\Sigma|}$ and say that a state transition is *deterministic* if $|\delta(q, x)| = 1$.

Topology changes. In contrast to the model presented in [13], our network model supports dynamic *topology changes* that belong to the following four classes:

1. *Edge deletion:* Remove a selected edge $e = \{u, v\}$ from the current graph. The corresponding ports $\phi_u(v)$ and $\phi_v(u)$ are removed and the messages stored in them are erased.
2. *Edge insertion:* Add an edge connecting nodes $u, v \in V$ to the current graph. New ports $\phi_u(v)$ and $\phi_v(u)$ are introduced storing the initial letter $\sigma_0 \in \Sigma$.
3. *Node deletion:* Remove a selected node $v \in V$ from the current graph with all its incident edges. The corresponding ports $\phi_v(u)$ of all v 's neighbors u are eliminated and the messages stored in them are erased.
4. *Node insertion:* Add a new isolated (i.e., with no neighbors) node to the current graph. Initially, the node resides in the initial state q_0 .

We assume that the schedule of these topology changes is controlled by an *oblivious adversary*. Formally, the strategy of the adversary associates a (possibly empty) set of topology changes with each round $r \in \mathbb{Z}_{>0}$ of the execution so that the total number of changes throughout the execution, denoted by C , is finite. This strategy may depend on the algorithm Π , but not on the random choices made during the execution.

To be precise, each round is divided into 4 successive steps as follows: (i) messages arrive at their destination ports; (ii) topology changes occur; (iii) the transition function is applied; and (iv) messages are transmitted. In particular, a message transmitted by node v in round r will not be read by node u in round $r+1$ if edge $\{u, v\}$ is inserted or deleted in round $r+1$. It is convenient to define the *adversarial graph sequence* $\mathcal{G} = G_1, G_2, \dots$ so that G_1 is the initial graph and G_{r+1} is the graph obtained from G_r by applying to it the topology changes scheduled for round r . By definition, \mathcal{G} is fully determined by the initial graph and the adversarial policy (and vice versa). The requirement that C is finite implies, in particular, that \mathcal{G} admits an infinite suffix of identical graphs. Let n be the largest number of nodes that co-existed in the same round, i.e., $n = \max_r |V(G_r)|$, where $V(G_r)$ is the node set of G_r .

Correctness. We say that node u resides in state q in round r if the state of u is q at the beginning of round r . The algorithm is said to be in an *output configuration* in round r if every node $u \in G_r$ resides in an output state (q_{yes} or q_{no}). The output configuration is said to be *correct* if the states of the nodes (treated as their output) correspond to a valid MIS of the current graph G_r . An algorithm Π is said to be *correct* if the following conditions are satisfied for every adversarial graph sequence: (C1) If Π is in a non-output configuration in round r , then it will move to an output configuration in some (finite) round $r' > r$ w.p. 1.¹ (C2) If Π is in an output configuration in round r , then this output configuration is correct (with respect to G_r). (C3) If Π is in an output configuration in round r and $G_{r+1} = G_r$, then Π remains in the same output configuration in round $r+1$.

¹ Throughout, we use w.p. to abbreviate “with probability” and w.h.p. to abbreviate “with high probability”, i.e., with probability n^{-c} for any constant c .

Restrictions on the Output States. Under the model of [13], the nodes are not allowed to change their output once they have entered an output state. On the other hand, this model allows for multiple output states that correspond to the same problem output (“yes” and “no” in the MIS case). In other words, it is required that the output states that correspond to each problem output form a sink of the transition function. Since our model accommodates dynamic topology changes which might turn a correct output configuration into an incorrect one, we lift this restriction: our model allows transitions from an output state to a non-output state, thus providing the algorithm designer with the possibility to escape output configurations that become incorrect. Nevertheless, to prevent nodes in an output state from taking any *meaningful* part in the computation process, we introduce the following new (with respect to [13]) restrictions: (1) each possible output is represented by a unique output state; (2) a transition from an output state to itself is never accompanied by a letter transmission (i.e., it transmits ε); (3) all transitions originating from an output state must be deterministic; and (4) a transition from an output state must lead either to itself or to a non-output state.

Runtime. Fix some adversarial graph sequence $\mathcal{G} = G_1, G_2, \dots$ and let η be an execution of a correct algorithm Π on \mathcal{G} (determined by the random choices of Π). Round r in η is said to be *silent* if Π is in a correct output configuration and no topology change occurs in round r . The *global runtime* of η is the number of non-silent rounds. Node v is said to be *active* in round r of η if it resides in a non-output state, i.e., some state in $Q - \{q_{yes}, q_{no}\}$. The *(local) runtime* of v in η is the number of rounds in which v is active.

Let $N_r(v)$ be the (inclusive) neighborhood of v and let $E_r(v)$ be the edges incident to v in G_r . Node v is said to be *affected* under \mathcal{G} if either v or one of its neighbors experienced an edge insertion/deletion, that is, there exists some round r and some node $u \in N_r(v)$ such that $E_r(u)$ is not identical to $E_{r+1}(u)$. Algorithm Π is said to be *effectively confining* if the following conditions are satisfied for every adversarial graph sequence \mathcal{G} : (1) the expected maximum local runtime of the non-affected nodes is $\log^{\mathcal{O}(1)} n$; and (2) the expected global runtime is $(C+1) \cdot \log^{\mathcal{O}(1)} n$, namely, $\log^{\mathcal{O}(1)} n$ when amortized over the number of topology changes. Notice that the bound on the global runtime directly implies the same bound on the local runtime of any affected node.

3 An MIS Algorithm

Our main goal is to design an algorithm under the Stone Age model for the maximal independent set (MIS) problem that is able to tolerate topology changes. For a graph $G = (V, E)$, a set of nodes $I \subseteq V$ is independent if for all $u, v \in I$, $\{u, v\} \notin E$. An independent set I is maximal if there is no other set $I' \subseteq V$ such that $I \subset I'$ and I' is independent.

Following the terminology introduced in the model section, we show that our algorithm is effectively confining. In Sect. 5, we provide a straightforward

lower bound example that shows that under our model, our solution is within a polylogarithmic factor from optimal. In other words, the linear dependency on the number of changes is inevitable. Throughout, the bounding parameter of our algorithm is 1. This indicates that a node is able to distinguish between the cases of having a single appearance of a letter σ in some of its ports and not having any appearances of σ in any of its ports.

The basic idea behind our algorithm is that we first use techniques from [13] to come up with an MIS quickly and then we fix any errors that the topology changes might induce. In other words, our goal is to first come up with a *proportional* MIS, where the likelihood of a node to join the MIS is inversely proportional to the number of neighbors the node has that have not yet decided their output. We partition the state set of our algorithm into two components. One of the components contains the input state and is responsible for computing the proportional MIS. Once a node has reached an output state or detects that it has been affected by a topology change, it transitions to the second component, which is responsible for fixing the errors, and never enters an active state of the first component. One of the reasons behind dividing our algorithm into two seemingly similar components is that the first component has stronger runtime guarantees, but requires that the nodes start in a “nice” configuration, whereas the second component does not have this requirement while providing weaker runtime bounds.

3.1 The Proportional Component

The component that computes the proportional MIS, called **Proportional**, follows closely the design from [13]. The goal of the rest of the section is to introduce a slightly modified version of their algorithm that allows the non-affected nodes to ignore affected nodes while constructing the initial MIS.

Proportional consists of states $Q = \{S, D_1, D_2, U_0, U_1, U_2, W, L\}$, where $Q_a = Q - \{W, L\}$ are referred to as *active* states and $q_{yes} = W$ and $q_{no} = L$ as *passive* states. We set S as the initial state. The communication alphabet is identical to the set of states; the algorithm is designed so that node v transmits letter q in round r whenever it resides in state q in round $r + 1$, i.e., state q was returned by the application of the transition function in round r .

Let us denote a state transition between states q and q' by $q \rightarrow q'$ omitting the rules associated with this transition. To ease our notation and to make our illustrations more readable, we say that each state transition $q \rightarrow q'$ is *delayed* by a set $\mathcal{D} = \mathcal{D}(q \rightarrow q') \subseteq Q$ of delaying states. For $q \rightarrow q'$ the set of delaying states corresponds to the states in $Q - \{q\}$ from which there is a state transition to q . Transition $q \rightarrow q'$ being delayed by state q'' indicates that node v does not execute transition $q \rightarrow q'$ as long as there is at least one letter in its ports that corresponds to the state q'' . We say that node v in state q is delayed if there is a neighbor w of v that resides in a state that delays at least one of the transitions from q , i.e., node v cannot execute some transition because of node w .

The main idea of **Proportional** is that each node v iteratively competes against its neighbors and the winner of a competition enters the MIS. Every

competition, for an active node v and its active neighbors, begins by the nodes entering state U_0 . During each round every node in state $U_j, j \in \{0, 1, 2\}$, assuming that it is not delayed, tosses a fair coin and proceeds to $U_{j+1 \bmod 3}$ if the coin shows heads and to D_2 otherwise. Notice that a competition does not necessarily start in the same round for two neighboring nodes. The circular delaying logic of the U -states ensure that node v is always at most one coin toss ahead of its neighbors. If node v observes that it is the only node in its neighborhood in a U -state, it enters state W , which corresponds to joining the MIS. In the case where, due to unfortunate coin tosses, v moves to state D_2 along with all of its neighbors, node v restarts the process by entering state D_1 .

We call a maximal contiguous sequence of rounds v spends in state $q \in Q_a$ a q -turn and a maximal contiguous sequence of turns starting from a D_1 -turn and not including another D_1 -turn a *tournament*. We index the tournaments and the turns within the tournament by positive integers. The correctness of **Proportional** in the case of no topology changes follows the same line of arguments as in [13].

We note that dynamically adding edges between nodes in states U_0, U_1 , and U_2 could potentially cause a deadlock. To avoid this, we add a state transition, that is not delayed by any state, from each state $Q - \{S, L\}$ to state D' (that is part of the fixing component explained in Sect. 3.2) under the condition that a node reads the initial letter S in at least one of its ports. In other words, node v enters state D' if an incident edge is added in any round $r > 1$. Refer to Fig. 1 for a detailed, though somewhat cluttered, illustration. If node v transitions to state D' due to this condition being met or if v is deleted while being in an active state, we say that v is *excluded* from the execution of **Proportional**.

3.2 The Greedy Component

Now we extend **Proportional** to fix the MIS in the case that a topology change leaves the network in an illegal state. This extension of **Proportional** is referred to as the *greedy component* and denoted by **Greedy**.

Intuition spotlight: *The basic idea is that nodes in states L and W verify locally that the configuration corresponds to a legal state of the network. If a node detects that the local configuration does not correspond to an MIS anymore, it revokes its output and tries again to join the MIS according to a slightly modified competition logic. The crucial difference is that we design Greedy without the circular delay logic. The reason behind the design is twofold: First, we cannot afford long chains of nodes being delayed. Second, since a dynamic change is only allowed to affect its 1-hop neighborhood, we cannot maintain the local synchrony similarly to Proportional.*

More precisely, the state set of the algorithm is extended by $Q_2 = \{U', D'\}$, where states U' or D' are referred to as active. To detect an invalid configuration, we add the following state transitions from the output states W and L : a transition from state L to state D' in case that a node v resides in state L and does not have any letters W in its ports and a transition from state W to

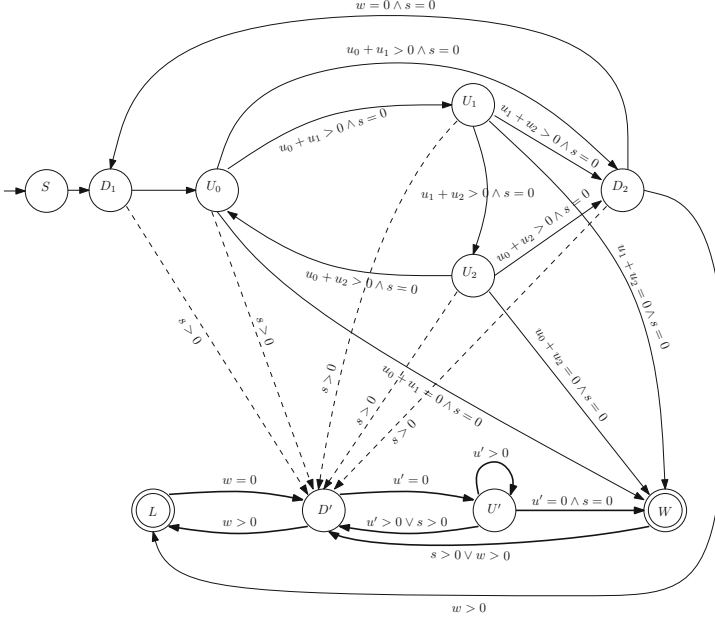


Fig. 1. The transition function of our MIS algorithm. The dashed lines correspond to the transitions that exclude nodes from the execution of **Proportional**. A state transition $q \rightarrow q'$ is delayed by state q'' if there is a transition indicated by a dashed edge or a thin edge from state q'' to q . A bold edge from q to q' implies that $q' \rightarrow q''$ is not delayed by q . As an example, the state transition from D' to U' is not delayed by state L and conversely, the state transition from D' to U' is delayed by state D_1 . Similarly to the other state machine illustrations, the lower case letters in the transition rules correspond to the number of appearances of the corresponding letter and the rules associated with the delays are omitted for clarity. If a node v is in state q and none of the conditions associated with the transitions from q are satisfied (e.g., v is in state D' , has a neighbor in state U' , and no neighbors in state W), then v remains in state q . These self-loops are omitted from the picture for clarity.

state D' in case v resides in state W and reads a letter W in its ports. Finally, to prevent nodes in the active states of **Proportional** and **Greedy** from entering state W at the same time and thus, inducing an incorrect output configuration, we add a transition from W and U' to D' in case v reads the initial letter S .

The logic of the new states U' and D' is the following: node v in state D' goes into state U' if it does not read the letter U' in its ports. Then u and its neighbors that transitioned from D' to U' during the same round compete against each other: In every round, v tosses a fair coin and goes back to state D' if the coin shows tails. We emphasize that nodes in state U' are not delayed by nodes in state D' . Then, if v remains the only node in its neighborhood in state U' , it declares itself as a winner and moves to state W . Conversely, if a neighbor of v wins, v goes into state L .

Observation 1 (*Proof deferred to the full version*). *A non-affected node is never in an active state of Greedy.*

4 Analysis

The authors of [13] analyze an algorithm very similar to the one induced by **Proportional**, showing that its runtime (on a static graph) is $O(\log^2 n)$ in expectation and w.h.p. In their analysis, they prove that with a positive constant probability, the number of edges decreases exponentially from one tournament to the next.

Intuition spotlight: *The dynamic nature of the graph \mathcal{G} prevents us from analyzing the execution of **Proportional** in the same manner. To overcome this obstacle, we introduce an auxiliary execution \mathcal{E} of **Proportional** that turns out to be much easier to analyze, yet serves as a good approximation for the real execution.*

Consider some execution η of **Proportional** on \mathcal{G} and recall that η is determined by the adversarial graph sequence \mathcal{G} and by the coin tosses of the nodes. We associate with η the *auxiliary execution* $\mathcal{E} = \mathcal{E}(\eta)$ obtained from η by copying the same coin tosses (for each node) and modifying the adversarial policy by applying to it the following two rules for $i = 1, 2, \dots$: (i) If node u is *excluded* during tournament i under η , then, instead, we *delete* node u immediately following its last U -turn under $\mathcal{E}(\eta)$. (ii) If node u becomes passive during tournament i under η and it would have remained active throughout tournament i under $\mathcal{E}(\eta)$, then we also *delete* node u immediately following its last U -turn under $\mathcal{E}(\eta)$.

Let $V_\eta(i)$ and $V(i)$ be the sets of nodes for which tournament i exists under η and $\mathcal{E}(\eta)$, respectively. Let $X^v(i)$ and $Y^v(i)$ denote the number of U -turns of node v in tournament i under η and $\mathcal{E}(\eta)$, respectively; to ensure that $X^v(i)$ and $Y^v(i)$ are well defined, we set $X^v(i) = 0$ if $v \notin V_\eta(i)$ and $Y^v(i) = 0$ if $v \notin V(i)$, i.e., if tournament i does not exist for v under η and $\mathcal{E}(\eta)$, respectively. Let $\Gamma_\eta(v, i)$ and $\Gamma(v, i)$ denote the (exclusive) neighborhood of v at the beginning of tournament i under η and $\mathcal{E}(\eta)$, respectively. Observe that by definition, $\Gamma_\eta(v, i) \subseteq V_\eta(i)$ and $\Gamma(v, i) \subseteq V(i)$. We say that node v *wins* in tournament i under η if $X^v(i) > X^w(i)$ for all $w \in \Gamma_\eta(v, i)$ and if v is not excluded in tournament i ; likewise, we say that node v *wins* in tournament i under $\mathcal{E}(\eta)$ if $Y^v(i) > Y^w(i)$ for all $w \in \Gamma(v, i)$ and if v is not deleted in tournament i . The following lemma plays a key role in showing that the number of tournaments in $\mathcal{E}(\eta)$ is at least as large as that in η .

Lemma 2 (*Proof deferred to the full version*). *If $v \in V(i)$ wins in tournament i under $\mathcal{E}(\eta)$, then v wins in tournament i under η .*

4.1 Runtime of Proportional

Next, we analyze the number of tournaments in $\mathcal{E}(\eta)$. Once we have a bound on the number of tournaments executed in $\mathcal{E}(\eta)$, it is fairly easy to obtain the

same bound for η . Similarly as before, the set of nodes for which tournament i exists according to $\mathcal{E}(\eta)$ is denoted by $V(i)$. Furthermore, let $\Gamma(v, i)$ be the set of neighbors of v for which tournament i exists in $\mathcal{E}(\eta)$ and let $E(i) = \bigcup_{v \in V(i)} \{\{v, w\} \mid w \in \Gamma(v, i)\}$ and $G(i) = (V(i), E(i))$. Notice that $G(i)$ is defined for the sake of the analysis and the topology of the underlying graph does not necessarily correspond to G_i in any round i , where G_i is the i th element in the adversarial graph sequence.

According to the design of **Proportional**, no node that once enters a passive state can enter an active state of **Proportional**. Furthermore, if any edge is added adjacent to node v after v has transitioned out of state S , node v will not participate in any tournament after the addition. Therefore, we get that $V(i+1) \subseteq V(i)$ and $E(i+1) \subseteq E(i)$ for any i . The following lemma plays a crucial role in the runtime analysis of **Proportional**.

Lemma 3 (*Proof deferred to the full version*). *There are two constant $0 < p, \ell < 1$ such that $|E(i+1)| \leq \ell |E(i)|$ with probability p .*

Theorem 4 (*Proof deferred to the full version*). *In expectation and w.h.p., any node participates in $\mathcal{O}(\log n)$ tournaments before becoming passive.*

The last step of the analysis of **Proportional** is to bound the number of rounds that any node v spends in an active state. Let $V_{\text{MAX}}(v)$ denote the maximal connected component of nodes in active states of **Proportional** that contains node v . Consider the following modification of **Proportional**: before starting tournament $i+1$, node v waits for every other node in $V_{\text{MAX}}(v)$ to finish tournament i or to become excluded. Notice that any node that gets inserted into the graph after node v has exited state S will not be part of $V_{\text{MAX}}(v)$. We do not claim that we know how to implement such a modification, but clearly the modified process is not faster than the original one.

The length of tournament i is determined by $\max_v \{X^v(i)\}$. Given that the random variables $X^v(i)$ are independent and follow the $\text{Geom}(1/2)$ distribution, we get that $\max_v \{X^v(i)\} \in \mathcal{O}(\log n)$ with high probability and in expectation. Combining with Observation 1, we get the following corollaries.

Corollary 5. *Let $t_v \geq 1$ be the round in which node v is inserted into the graph, where $t_v = 1$ if $v \in G_1$. If v is not deleted, it enters either state W or L by time $t_v + \mathcal{O}(\log^2 n)$ w.h.p. and in expectation.*

Corollary 6. *The maximum runtime of any non-affected node is $\mathcal{O}(\log^2 n)$ in expectation and w.h.p.*

4.2 The Quality of an MIS

Before we go to the analysis of **Greedy**, we want to point out that even though an invalid configuration can be detected locally, a single topology change can have an influence in an arbitrary subgraph.

We begin the analysis of **Greedy** by taking a closer look at the properties of non-affected nodes in the passive states and show that if a node has a high

degree, it is either unlikely for this node to be in the MIS or that the neighbors of this node have more than one MIS node in their neighborhoods. Let i be the index of the tournament in which node v enters state W . We say that node v *covers* node $w \in \Gamma(v, i) \cup \{v\}$ if w entered state L in tournament i .

Definition 7. *The quality $q(v)$ of node v is given by $q(v) = |\{w \in \Gamma(v, i) \cup \{v\} \mid v \text{ covers } w\}|$. The quality $q(B)$ of any set of nodes B is defined as $\sum_{v \in B} q(v)$.*

Lemma 8 (Proof deferred to the full version). $\mathbb{E}[q(v)] \in \mathcal{O}(\log n)$ for any node v .

The quality of a node v gives an upper bound on the number of nodes in the neighborhood of v that are only covered by v . Let v be a node in state L . If t is the first round such that there are no nodes adjacent to v in state W , we say that v is *released* at time t . Similarly, we say that node v in state W is released if an adjacent edge is added. Every node can only be released once, i.e., node v counts as released even if it eventually again has a neighboring node in state W or if it enters state W .

Lemma 9 (Proof deferred to the full version). Let H be the set of nodes that are eventually released. Then $\mathbb{E}[|H|] \in \mathcal{O}(C \log n)$.

4.3 Fixing the MIS

We call a maximal contiguous sequence of rounds in which node v resides in state U' a *greedy tournament*. Unlike with **Proportional**, we index these greedy tournaments by the time this particular tournament starts. In other words, if node v resides in state D' in round $t - 1$ and in state U' in round t , we index this tournament by t . We denote the random variable that counts the number of transitions from U' to itself in greedy tournament t by node v by $X^v(t)$. Notice that $X^v(t)$ obeys $\text{Geom}(1/2)$ unless v is deleted or an adjacent edge is added during greedy tournament t . We say that a greedy tournament t is *active* if there is at least one node v in state U' of greedy tournament t .

Observation 10 (Proof deferred to the full version). There are at most $\mathcal{O}(\log n)$ rounds in which greedy tournament t is active in expectation and w.h.p.

Lemma 11 (Proof deferred to the full version). The total expected number of greedy tournaments is $\mathcal{O}(C \log n)$.

Observation 12 (Proof deferred to the full version). Let k be the total number of greedy tournaments. There are at most $2k + 2C$ non-silent rounds without either (i) at least one node in an active state of **Proportional** or (ii) an active greedy tournament.

Next we show that our MIS algorithm, that consists of **Proportional** and **Greedy**, fulfills the correctness properties given in the model section. Then, we establish that our MIS algorithm is indeed effectively confining.

Lemma 13 (*Proof deferred to the full version*). *Our MIS algorithm is correct.*

Theorem 14 (*Proof deferred to the full version*). *The global runtime of our MIS algorithm is $\mathcal{O}((C+1)\log^2 n)$ in expectation.*

5 Lower Bound

The runtime of our algorithm might seem rather slow at the first glance, since it is linear in the number of topology changes. In this section, we show that one cannot get rid of the linear dependency, i.e., there are graphs where the runtime of any algorithm grows at least linearly with the number of changes. In particular, we construct a graph where the runtime of any algorithm is $\Omega(C)$.

Let G^ℓ be a graph that consists of n nodes and $n = 3\ell$. The graph consists of ℓ components B_i , where B_i is a 3-clique for each $1 \leq i \leq \ell$. In addition, let u_1, \dots, u_ℓ be a set of nodes such that $u_i \in B_i$ for every i .

Theorem 15. *There exists a graph G and an schedule of updates such that the runtime of any algorithm on G is $\Omega(C)$ in expectation and w.h.p.*

Proof. Consider graph G^ℓ and the following adversarial strategy. In round $2i - 1 \geq 1$, the adversary deletes node u_i , i.e., one of the nodes in component B_i . Our goal is to show that at least a constant fraction of the first $2C$ rounds are non-silent for any $C \leq n/3$. Consider round $2i$ and component B_i . Since the nodes in component B_i form a clique, their views are identical. Therefore, according to any algorithm that computes an MIS, their probability to join the MIS is equal, i.e., $1/3$. Thus, the probability that the nodes in $B_i - \{u_i\}$ are not in the MIS in any round $2j \leq 2i$ is at least $1/3$.

Let X_i then be the indicator random variable, where $X_i = 0$ if round i is silent and 1 otherwise and let $X = \sum_{i=1}^{\infty} X_i$ be the random variable that counts the number of non-silent rounds. Since any round $1 \leq 2i \leq 2C$ is non-silent with at least probability $1/3$, we get that $\mathbb{E}[X] \geq (1/3)C$. Now by applying a Chernoff bound, we get that $\mathbb{P}[X < 1/2\mathbb{E}[X]] = \mathbb{P}[X < (1/2) \cdot (1/3)C] \leq 2^{-C/12}$. Since $C = n/3$, we get that $\mathbb{P}[X < 1/2\mathbb{E}[X]] \in \mathcal{O}(n^{-k})$ for any constant k and thus, the claim follows.

References

1. Afek, Y., Alon, N., Bar-Joseph, Z., Cornejo, A., Haeupler, B., Kuhn, F.: Beeping a maximal independent set. In: Peleg, D. (ed.) DISC 2011. LNCS, vol. 6950, pp. 32–50. Springer, Heidelberg (2011). doi:[10.1007/978-3-642-24100-0_3](https://doi.org/10.1007/978-3-642-24100-0_3)
2. Afek, Y., Alon, N., Barad, O., Hornstein, E., Barkai, N., Bar-Joseph, Z.: A biological solution to a fundamental distributed computing problem. *Science* **331**(6014), 183–185 (2011)
3. Alon, N., Babai, L., Itai, A.: A fast and simple randomized parallel algorithm for the maximal independent set problem. *J. Algorithms* **7**, 567–583 (1986)

4. Angluin, D., Aspnes, J., Diamadi, Z., Fischer, M.J., Peralta, R.: Computation in networks of passively mobile finite-state sensors. *Distrib. Comput.* **18**(4), 235–253 (2006)
5. Aspnes, J., Ruppert, E.: An introduction to population protocols. In: Garbinato, B., Miranda, H., Rodrigues, L. (eds.) *Middleware for Network Eccentric and Mobile Applications*, pp. 97–120. Springer, Heidelberg (2009)
6. Awerbuch, B., Patt-Shamir, B., Peleg, D., Saks, M.: Adapting to asynchronous dynamic networks (extended abstract). In: *Proceedings of the 24th Annual ACM Symposium on Theory of Computing (STOC)*, pp. 557–570 (1992)
7. Awerbuch, B., Sipser, M.: Dynamic networks are as fast as static networks. In: *29th Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 206–220 (1988)
8. Barenboim, L., Elkin, M., Pettie, S., Schneider, J.: The locality of distributed symmetry breaking. In: *Proceedings of the 53rd Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 321–330 (2012)
9. Chlamtac, I., Kutten, S.: On broadcasting in radio networks-problem analysis and protocol design. *Commun., IEEE Trans. Commun.* **33**(12), 1240–1246 (1985)
10. Cole, R., Vishkin, U.: Deterministic coin tossing with applications to optimal parallel list ranking. *Inf. Control* **70**(1), 32–53 (1986)
11. Cornejo, A., Kuhn, F.: Deploying wireless networks with beeps. In: Lynch, N.A., Shvartsman, A.A. (eds.) *DISC 2010. LNCS*, vol. 6343, pp. 148–162. Springer, Heidelberg (2010). doi:[10.1007/978-3-642-15763-9_15](https://doi.org/10.1007/978-3-642-15763-9_15)
12. Daum, S., Ghaffari, M., Gilbert, S., Kuhn, F., Newport, C.: Maximal independent sets in multichannel radio networks. In: *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, pp. 335–344 (2013)
13. Emek, Y., Wattenhofer, R.: Stone age distributed computing. In: *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, pp. 137–146 (2013)
14. Fisher, J., Henzinger, T.A., Mateescu, M., Piterman, N.: Bounded asynchrony: concurrency for modeling cell-cell interactions. In: Fisher, J. (ed.) *FMSB 2008. LNCS*, vol. 5054, pp. 17–32. Springer, Heidelberg (2008). doi:[10.1007/978-3-540-68413-8_2](https://doi.org/10.1007/978-3-540-68413-8_2)
15. Flury, R., Wattenhofer, R.: Slotted programming for sensor networks. In: *IPSN* (2010)
16. Gardner, M.: The fantastic combinations of John Conway’s new solitaire game ‘life’. *Sci. Am.* **223**(4), 120–123 (1970)
17. Ghaffari, M.: An improved distributed algorithm for maximal independent set. In: *Proceedings of the 27th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 270–277 (2016)
18. Ghosh, S., Gupta, A., Herman, T., Pemmaraju, S.: Fault-containing self-stabilizing algorithms. In: *Proceedings of the 15th Annual ACM Symposium on Principles of Distributed Computing (PODC)*, pp. 45–54 (1996)
19. Hayes, T., Saia, J., Trehan, A.: The forgiving graph: a distributed data structure for low stretch under adversarial attack. In: *Proceedings of the 28th ACM Symposium on Principles of Distributed Computing (PODC)*, pp. 121–130 (2009)
20. König, M., Wattenhofer, R.: On local fixing. In: Baldoni, R., Nisse, N., Steen, M. (eds.) *OPDIS 2013. LNCS*, vol. 8304, pp. 191–205. Springer, Heidelberg (2013). doi:[10.1007/978-3-319-03850-6_14](https://doi.org/10.1007/978-3-319-03850-6_14)
21. Korman, A.: Improved compact routing schemes for dynamic trees. In: *Proceedings of the 27th Annual ACM Symposium on Principles of Distributed Computing (PODC)*, pp. 185–194 (2008)

22. Kuhn, F., Moscibroda, T., Wattenhofer, R.: What cannot be computed locally! In: Proceedings of the 23th Annual ACM Symposium on Principles of Distributed Computing (PODC), pp. 300–309 (2004)
23. Kuhn, F., Schmid, S., Wattenhofer, R.: A self-repairing peer-to-peer system resilient to dynamic adversarial churn. In: Castro, M., Renesse, R. (eds.) IPTPS 2005. LNCS, vol. 3640, pp. 13–23. Springer, Heidelberg (2005). doi:[10.1007/11558989_2](https://doi.org/10.1007/11558989_2)
24. Kutten, S., Peleg, D.: Fault-local distributed mending. *J. Algorithms* **30**(1), 144–165 (1999)
25. Kutten, S., Peleg, D.: Tight fault locality. *SIAM J. Comput.* **30**(1), 247–268 (2000)
26. Li, X., Misra, J., Plaxton, C.G.: Active and concurrent topology maintenance. In: Guerraoui, R. (ed.) DISC 2004. LNCS, vol. 3274, pp. 320–334. Springer, Heidelberg (2004). doi:[10.1007/978-3-540-30186-8_23](https://doi.org/10.1007/978-3-540-30186-8_23)
27. Lin, J.C., Huang, T.: An efficient fault-containing self-stabilizing algorithm for finding a maximal independent set. *IEEE Trans. Parallel Distrib. Syst.* **14**(8), 742–754 (2003)
28. Linial, N.: Locality in distributed graph algorithms. *SIAM J. Comput.* **21**(1), 193–201 (1992)
29. Luby, M.: A simple parallel algorithm for the maximal independent set problem. *SIAM J. Comput.* **15**, 1036–1055 (1986)
30. Malpani, N., Welch, J.L., Waidya, N.: Leader election algorithms for mobile ad hoc networks. In: Proceedings of the 4th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIAL-M), pp. 96–103 (2003)
31. Marr, C., Hütt, M.T.: Outer-totalistic cellular automata on graphs. *Phys. Lett. A* **373**(5), 546–549 (2009)
32. Michail, O., Chatzigiannakis, I., Spirakis, P.G.: New Models for Population Protocols. *Synthesis Lectures on Distributed Computing Theory*. Morgan & Claypool Publishers, San Rafael (2011)
33. von Neumann, J.: *Theory of Self-Reproducing Automata*. University of Illinois Press, Champaign (1966)
34. Panconesi, A., Srinivasan, A.: On the complexity of distributed network decomposition. *J. Algorithms* **20**(2), 356–374 (1996)
35. Peleg, D.: *Distributed Computing: A Locality-Sensitive Approach*. Society for Industrial and Applied Mathematics, Philadelphia (2000)
36. Schneider, J., Wattenhofer, R.: A log-star distributed maximal independent set algorithm for growth-bounded graphs. In: Proceedings of the 27th ACM Symposium on Principles of Distributed Computing (PODC), pp. 35–44 (2008)
37. Scott, A., Jeavons, P., Xu, L.: Feedback from nature: an optimal distributed algorithm for maximal independent set selection. In: Proceedings of the 32nd Symposium on Principles of Distributed Computing (PODC), pp. 147–156 (2013)
38. Shukla, S., Rosenkrantz, D., Ravi, S.S.: Observations on self-stabilizing graph algorithms for anonymous networks (extended abstract). In: Proceedings of the Second Workshop on Self-Stabilizing Systems, pp. 1–15 (1995)
39. Valiant, L.G.: Parallel computation. In: 7th IBM Symposium on Mathematical Foundations of Computer Science (1982)
40. Walter, J., Welch, J.L., Vaidya, N.: A mutual exclusion algorithm for ad hoc mobile networks. *Wireless Netw.* **7**, 585–600 (1998)
41. Wolfram, S.: *A New Kind of Science*. Wolfram Media, Champaign (2002)

Structural Information and Communication Complexity
23rd International Colloquium, SIROCCO 2016, Helsinki,
Finland, July 19-21, 2016, Revised Selected Papers
Suomela, J. (Ed.)
2016, XXIX, 408 p. 51 illus., Softcover
ISBN: 978-3-319-48313-9