Inf. Process. Lett. 188 (2025) 106531

Contents lists available at ScienceDirect

Information Processing Letters

journal homepage: www.elsevier.com/locate/ipl

Luby's MIS algorithms made self-stabilizing

George Giakkoupis [a], Volker Turau [b,*], Isabella Ziccardi [c]

[a] Inria, Univ. Rennes, CNRS, IRISA, Rennes, France
[b] Institute of Telematics, Hamburg University of Technology, Hamburg, Germany
[c] Bocconi University, BIDSA, Milan, Italy

ARTICLE INFO

Keywords:
Self-stabilization
Fault tolerance
Distributed computing
Maximal independent set (MIS)

ABSTRACT

We reconsider two well-known distributed randomized algorithms computing a maximal independent set, proposed in the seminal work of Luby (1986). We enhance these algorithms such that they become self-stabilizing without sacrificing their run-time, i.e., both stabilize in $O(\log n)$ synchronous rounds with high probability on any n-node graph. The first algorithm gets along with three states, but needs to know an upper bound on the maximum degree. The second does not need any information about the graph, but uses a number of states that is linear in the node degree. Both algorithms use messages of logarithmic size.

1. Introduction

A major challenge for large-scale systems is to handle fault recovery. One way to accommodate this challenge is by employing self-stabilizing algorithms [9,10]. These algorithms guarantee that starting from an arbitrary state, the system will eventually reach a correct state, and it will stay in a correct state as long as no faults occur. Thus, the system can recover from transient faults corrupting the state of a node without external intervention.

We consider the maximal independent set (MIS) problem — a fundamental problem in parallel and distributed computing. An MIS of a graph $G = (V, E)$ is a subset of vertices $I \subseteq V$, such that no two vertices in I are adjacent to each other and I is maximal with respect to inclusion. The significance of the problem was first recognized in the 1980s, due to its importance in symmetry breaking [3,18], and has been extensively studied ever since (see survey [7] for an overview of works until 2015, and [5,12] for recent state-of-the-art results).

Several self-stabilizing MIS algorithms have been proposed. The first such algorithms were deterministic, relying on unique node IDs to break symmetry [14,16,21,22]. All these algorithms used just a constant number of states per node, and their stabilization time was $O(n)$ or larger, where n is the number of nodes. Their notion of time varied depending on the model used: sequential, synchronous, or a more general adversarial distributed model (see [15] for a survey of models and results). Recently, Barenboim at al. [6] proposed the first sub-linear deterministic self-stabilizing synchronous MIS algorithm, stabilizing in $O(\Delta + \log^* n)$ rounds, where Δ is the maximum degree, using a number of states polynomial in Δ.

A few randomized self-stabilizing MIS algorithms assuming very weak communication models, such as the beeping model [8], have also been proposed [1,11,13]. The MIS algorithm for the beeping model proposed in [1] stabilizes in $O(\log^3 n)$ rounds with high probability (w.h.p.), using a polynomial number of states, provided each node knows an upper bound for n. A constant-state algorithm for the beeping model, proposed in [13], was shown to stabilize in a poly-logarithmic number of rounds w.h.p., but only for specific families of graphs. Finally, assuming a slightly stronger communication model than beeping, a constant-state algorithm proposed in [11] stabilizes in $O(\log^2 n)$ rounds w.h.p. on constant-diameter graphs.

Non-self-stabilizing MIS algorithms can be made self-stabilizing by using general schemes for transforming non-self-stabilizing synchronous algorithms to asynchronous self-stabilizing ones [2,4,17]. However, these transformations may result in a significant growth in the run-time, the message size, or the number of states. In addition, they only work for deterministic algorithms.

In this paper, we present two new randomized self-stabilizing synchronous MIS algorithms, based on *Luby's MIS algorithms* [18]. Among the most elegant and well-known distributed graph algorithms, Luby's celebrated MIS algorithms[1] were originally presented in the parallel (PRAM) model, but can be directly translated to distributed algorithms in the failure-free synchronous message-passing model (see, e.g., [19,20]).

* Corresponding author.
E-mail address: turau@tuhh.de (V. Turau).

[1] Luby was awarded the 2016 Dijkstra Prize in Distributed Computing for his paper [18].

https://doi.org/10.1016/j.ipl.2024.106531
Received 26 November 2023; Received in revised form 9 August 2024; Accepted 9 August 2024
Available online 13 August 2024

The self-stabilizing variants we propose maintain the run-time of the original algorithms, stabilizing in $O(\log n)$ rounds w.h.p. The first one uses a constant number of states, and requires an upper bound Δ' on the maximum degree Δ. The second algorithm does not need any global information about the graph, and uses a number of states that is linear in the node's degree. Both algorithms use messages of logarithmic size. Precisely, in the first algorithm messages have size $O(\log \Delta')$ bits, while in the second they have size logarithmic in the degree of the sender node, and thus have size $O(\log \Delta)$ bits.

Unlike the existing randomized self-stabilizing MIS algorithms [1, 11,13], which assume a very weak communication model but have super logarithmic stabilization times, the algorithms we propose assume a more powerful model but stabilize in logarithmic time.

2. Two self-stabilizing MIS algorithms

We propose two the self-stabilizing MIS algorithms, Algorithm 1 and Algorithm 2. The algorithms operate in the synchronous message-passing model, where in each synchronous round all nodes in parallel: send messages to their neighbors, then read the messages received, and finally update their state. In the state-machine formulation (see, e.g., [19, Chapter 2]), the algorithm specifies for each node the set of possible states, the message alphabet, the message-generation function, which is a function of the node's state, and the state-transition function, which is a function of the node's state and the received messages. In our algorithms, nodes have no IDs, thus they are identical state machines. Also their message-generation function is probabilistic, and the same message is sent to all neighbors of a node in the same round.

The state-machine formulation allows for a refined description of the space requirements of the algorithm. In particular, it distinguishes between the node's state, which is the information a node must store from one round to the next, the storage of messages sent and received in a round, and the space of 'temporary' variables used in a round to implement the message-generation and state transition functions. These temporary variables are implementation-specific, and their values do not persist beyond a single round. The formal state-machine description of our algorithms can be easily deduced from their pseudo-code description.

For each node, both our algorithms maintain a state variable *status* that can assume values *active*, *in*, and *out*. This is the only state that Algorithm 1 needs to maintain from one round to the next, while Algorithm 2 uses an additional variable d (discussed later).

In both algorithms, starting from an arbitrary state, each node changes its status at most once after the first round. Active nodes try to enter the MIS in a contention-based style, and eventually nodes with status *in* form an MIS and all other nodes have status *out*. Symmetry breaking between active nodes is based on random values, r, that active nodes generate independently in each round. In Algorithm 1, the generation of values r is governed by global parameter Δ', which is an upper bound on the maximum degree; while in Algorithm 2, it is governed by the number of active neighbors of each node. For that, Algorithm 2 must maintain an additional local variable d storing the number of active neighbors. Hence, in Algorithm 1 each node has three states, while in Algorithm 2 each node has a number of states linear in its degree.

Luby's MIS algorithms operate in phases of a fixed number of rounds. In Algorithms 1 and 2, we dispense with the idea of phases, since the number of rounds required to synchronize the nodes' phases would be at least equal to the diameter of the graph. Giving up phases results in a 'stale information' problem, which manifests itself in two ways. The first one, which applies to both algorithms, is that a neighbor u of a node v joining the MIS in round t may prevent some other neighbor w of u from join the MIS in round t, even though u will change its status to *out* in the next round, $t+1$. The second problem, which applies only to Algorithm 2, is that variable d contains the number of active neighbors *from two rounds ago*, which may be much larger than the current number.

Algorithm 1: Self-stabilizing version of Luby's *Algorithm A* [18].

```
state: status ∈ {in, out, active}
in each round t = 1, 2, ... do
    if status = active then
        choose r unif. at random from 1, ..., 2Δ'
    else if status = in then r ← 0
    if status ∈ {active, in} then
        send message r to all neighbors
    receive messages sent by neighbors
    M ← set of received values of r
    update(M)
function update(M)
    if (status = in ∧ 0 ∈ M) ∨ (status = out ∧ 0 ∉ M) then status ← active
    else if status = active ∧ 0 ∈ M then
        status ← out
    else if status = active ∧ max(M ∪ {0}) < r then
        status ← in
```

Nevertheless, our refined analysis proves that both algorithms stabilize in $O(\log n)$ rounds w.h.p.

Algorithms 1 and 2 are not silent: All nodes in the MIS broadcast a single bit (zero) in each round, while all other nodes become silent eventually. It is easy to see that it is impossible to have a silent self-stabilizing MIS algorithm (for otherwise, we could initialize every node to a silent state such that the node is in the MIS).

Notation Let $G = (V, E)$ be a graph with n nodes. For each node $v \in V$, $N_G(v)$ denotes the set of v's neighbors in G, and $\deg_G(v) = |N_G(v)|$ is the degree of v. Also, for any set $X \subseteq V$, we define $N_G(X) = \bigcup_{v \in X} N_G(v)$. When it is clear from the context we omit subscript G from these notations. For $X \subseteq V$, $G[X]$ denotes the subgraph induced by X, and $E(X)$ is the set of edges of $G[X]$. By $E(v, X)$ we denote the set of edges incident to v in graph $G[X]$.

For both Algorithms 1 and 2, we use the following notation. For each node v, each *state* variable x (i.e., *status* or d), and any $t \geq 0$, we denote by $v.x_t$ the value of variable x of node v *at the end of round* t.[2] For *non-state* variables x (i.e., r and M), we write $v.x_t$, for $t \geq 1$, to denote the variable's value *during round* t. For $t \geq 0$, we let $I_t = \{v \in V : v.status_t = in\}$ be the set of node with status *in* at the end of round t. Similarly, O_t and A_t are the sets of nodes of status *out* and *active*, respectively, at the end of round t. Clearly, $V = I_t \cup O_t \cup A_t$.

By $\mathcal{F}_t$ we denote the filtration of the execution process up to the end of round t, which consists of the initial states of nodes and their random choices in each round $r \in \{1, \ldots, t\}$. Thus, from $\mathcal{F}_t$ we can compute the state of every node after each such round r.

2.1. Algorithm 1

The first self-stabilizing MIS algorithm we propose, Algorithm 1, is an adaptation of parallel *Algorithm A* proposed by Luby in [18] (see also *LubyMIS* algorithm described by Lynch [19] in the synchronous message-passing model). Algorithm A of [18] (adapted to the message-passing model) operates in phases of three rounds: In the first round each active node samples a uniformly random number r from a large enough range, then it broadcasts r to its active neighbors, and if its r value is larger than those of its neighbors' then it joins the MIS. In the second round nodes that have just joined the MIS inform their active neighbors, which remove themselves from the competition. In the third round newly removed nodes inform their neighbors, so that the remaining active nodes know which of their neighbors are still active.

[2] The 'end of round 0' refers to the initial configuration, before the first round.

Technically, the third round is not necessary, it just reduces the number of messages. All nodes are active initially.

The range from which uniformly random values r were sampled in Algorithm A in [18] was chosen for convenience to be $\{1,\dots,n^4\}$. We observe that it suffices to use a range of linear size in the maximum degree Δ (or in an upper bound Δ' of Δ). We thus use range $\{1,\dots,2\Delta'\}$ in Algorithm 1.

Algorithm 1 does not have phases, and nodes that have joined the MIS signal this *in every* future round by broadcasting special value 0. Thus, one round later, active nodes know if any of their neighbors are in the MIS, in which case they change their status to *out* and remain silent in future rounds. As mentioned earlier, this one-round delay introduces some additional complication to the complexity analysis.

The `update()` procedure of Algorithm 1 implements the rules by which nodes updates their status, as described above. Any inconsistencies, i.e., adjacent nodes of status *in*, or nodes of status *out* without neighbors in status *in*, are resolved in the first round, by the first rule of the `update()` procedure, and no new inconsistencies are introduced thereafter.[3]

Recall that I_t, O_t, and A_t denote the sets of node with status respectively *in*, *out*, and *active* at the end of round t.

Lemma 1. *Let $v \in V$.*

(a) For any $t \geq 1$, if $v \in I_t$ then $v \in I_{t+1}$ and $w \in O_{t+1}$ for all $w \in N(v)$.
(b) For any $t \geq 2$, if $v \in O_t$ then $v \in O_{t+1}$ and $w \in I_{t-1}$ for some $w \in N(v)$.

Proof. From the algorithm, for any $t \geq 0$, (i) if $v \in I_t$ and $w \notin I_t$ for all $w \in N(v)$ then $v \in I_{t+1}$; (ii) if $v \in I_t$ and $w \in I_t$ for some $w \in N(v)$ then $v \in A_{t+1}$; (iii) if $v \notin I_t$ and $w \in I_t$ for some $w \in N(v)$ then $v \in O_{t+1}$. Also, (iv) if $v \notin I_t$ and $v \in I_{t+1}$ then $w \notin I_{t+1}$ for all $w \in N(v)$, because $v \in A_t$ and $w \notin I_t$ for all $w \in N(v)$ and $v.r_{t+1} > w.r_{t+1}$ for all $w \in N(v) \cap A_t$, thus no $w \in N(v) \cap A_t$ can change its status to *in* in the same round, $t+1$.

Let $t \geq 1$ and suppose that $v \in I_t$. If $v \in I_{t-1}$ then from (ii), $w \notin I_{t-1}$ for all $w \in N(v)$, and from (iii), $w \in O_t$ for all $w \in N(v)$. Applying (i) and again (iii) gives $v \in I_{t+1}$ and $w \in O_{t+1}$ for all $w \in N(v)$.

If $v \in I_t$ but $v \notin I_{t-1}$ then from (iv), $w \notin I_t$ for all $w \in N(v)$. Then, as before, (i) gives $v \in I_{t+1}$, and (iii) gives $w \in O_{t+1}$ for all $w \in N(v)$.

Next let $t \geq 2$ and suppose that $v \in O_t$. Then, from the algorithm, $w \in I_{t-1}$ for some $w \in N(v)$, and since $t-1 \geq 1$ part (a) applied twice gives $v \in O_{t+1}$. □

From Lemma 1 we can immediately obtain the next corollary, which states that (ignoring the first couple of rounds) sets I_t and O_t can only increase over time (thus A_t can only decrease), and O_t is precisely the set of nodes with a neighbor in I_{t-1}; hence I_t is an MIS, when A_t becomes empty.

Corollary 2. *Let $t \geq 2$. Then $I_{t-1} \subseteq I_t$, $O_t \subseteq O_{t+1}$, $A_{t+1} \subseteq A_t$, and $O_t = N(I_{t-1})$. Also, if $A_t = \emptyset$ then I_t is an MIS.*

Theorem 3. *Algorithm 1 computes an MIS in $O(\log n)$ rounds w.h.p.*

Proof. Let $G_t = G[A_t]$ be the subgraph of G induced by A_t, and $E_t = E(A_t)$ be its set of edges. We will write $N_t(v)$ and $d_t(v)$ to denote $N_{G_t}(v)$ and $\deg_{G_t}(v)$, respectively. From Corollary 2, $A_{t+1} \subseteq A_t$ and thus $E_{t+1} \subseteq E_t$, for $t \geq 2$.

We prove that for all $t \geq 2$,

$$\mathbb{E}\left[\left|E_t \setminus E_{t+2}\right| \mid \mathcal{F}_t\right] \geq e^{-1} \cdot |E_t|/2,$$

[3] Note that it is not necessary to explicitly determine the set M passed as argument to `update()`. It suffices to know the largest received value r and whether value 0 was received.

Algorithm 2: Self-stabilizing version of Luby's *Algorithm B* [18].

```
state: status ∈ {in, out, active},  d ∈ {0, ..., deg(v)}
in each round t = 1, 2, ... do
    if status = active then
        r ← d + 1 with probability 1/max{2d,1}, and r ← 1 otherwise
    else if status = in then r ← 0
    if status ∈ {active, in} then
        send message r to all neighbors
    receive messages sent by neighbors
    M ← set of messages received
    d ← number of non-zero messages received
    update(M)                    // function of Algorithm 1
```

i.e., the expected number of edges between active nodes drops by at least a constant factor every *two* rounds. We use the idea of *preemptive removal* used in [20], which we adapt to the current setting. Let $X_t = A_t \cap N(I_t)$ and $Y_t = A_t \setminus N(I_t)$, and note that $X_t \subseteq N(I_t) = O_{t+1} \subseteq O_{t+2}$ by Corollary 2. For each $v \in Y_t$ and $u \in N(v) \cap Y_t$, we say that v *preemptively removes* u in round $t+1$, if $v.r_{t+1} > w.r_{t+1}$ for all $w \in N_t(v) \cup N_t(u) \setminus \{v\}$. Clearly, a node $u \in Y_t$ is preemptively removed by at most one neighbor in round $t+1$ (by definition only neighbors $v \in Y_t$ may preemptively remove u). If v preemptively removes u in round $t+1$ then $v \in I_{t+1}$, thus $u \in O_{t+2}$ and for all $w \in N(u)$ edge $\{u,w\}$ is not in E_{t+2}. Therefore, an edge $\{v,u\} \in E_t$ is not in E_{t+2}, if at least one of u, v belongs to Y_t and is preemptively removed in round $t+1$, or at least one of u, v belongs to X_t, since $X_t \subseteq O_{t+2}$ as we saw above.

Next we compute the probability that v preemptively removes u in round $t+1$, for any pair of $v \in Y_t$ and $u \in N(v) \cap Y_t$. The probability that $v.r_{t+1} \neq w.r_{t+1}$ for all $w \in N_t(v) \cup N_t(u) \setminus \{v\}$ is at least

$$\left(\frac{2\Delta'-1}{2\Delta'}\right)^{d_t(v)+d_t(u)-1} \geq \left(\frac{2\Delta'-1}{2\Delta'}\right)^{2\Delta'-1} \geq e^{-1}.$$

And given that this holds, the conditional probability that $v.r_{t+1}$ is the largest among these values is at least $1/(d_t(v)+d_t(u))$, due to symmetry and because there are at most $d_t(v)+d_t(u)$ distinct values. Therefore, the probability that v preemptively removes u in round $t+1$ is at least $e^{-1}/(d_t(v)+d_t(u))$.

Recall now that $\{v,u\} \in E_t \setminus E_{t+2}$ if u or v either belongs to Y_t and is preemptively removed in round $t+1$, or belongs to X_i. Since this condition may hold simultaneously for both endpoints u and v, we obtain that $2 \cdot \mathbb{E}\left[\left|E_t \setminus E_{t+2}\right| \mid \mathcal{F}_t\right]$ is at least

$$\sum_{v \in Y_t} \sum_{u \in N(v) \cap Y_t} \frac{e^{-1} \cdot d_t(v)}{d_t(v)+d_t(u)} + \sum_{v \in X_t} d_t(v) =$$

$$\sum_{\substack{\{v,u\} \in E_t: \\ v,u \in Y_t}} \left(\frac{e^{-1} \cdot d_t(v)}{d_t(v)+d_t(u)} + \frac{e^{-1} \cdot d_t(u)}{d_t(v)+d_t(u)}\right) +$$

$$\sum_{\substack{\{v,u\} \in E_t: \\ v \in X_t, u \in Y_t}} 1 + \sum_{\substack{\{v,u\} \in E_t: \\ v,u \in X_t}} 2 \geq \sum_{\{v,u\} \in E_t} e^{-1} = e^{-1} \cdot |E_t|.$$

Therefore, $\mathbb{E}\left[\left|E_t \setminus E_{t+2}\right| \mid \mathcal{F}_t\right] \geq e^{-1} \cdot |E_t|/2$, as desired.

It follows that $\mathbb{E}\left[\left|E_{t+2}\right|\right] \leq \delta\, \mathbb{E}\left[\left|E_t\right|\right]$, where $\delta = 1 - e^{-1}/2$. Applying this inequality repeatedly, we obtain for any even $t \geq 2$ that $\mathbb{E}\left[\left|E_t\right|\right] \leq \delta^{t/2-1}|E|$. Choosing $t = \Theta(\log(n|E|)/\log\delta) = \Theta(\log n)$ large enough, we obtain that $\mathbb{E}\left[\left|E_t\right|\right] \leq 1/\mathrm{poly}(n)$, and Markov's inequality yields that w.h.p. $|E_t| = 0$ and thus $A_{t+1} = \emptyset$, where the extra round deals with any isolated nodes left in G_t. The theorem then follows by applying Corollary 2. □

2.2. Algorithm 2

Our second self-stabilizing MIS algorithm, Algorithm 2, is based on parallel *Algorithm B* proposed by Luby in [18] (a similar algorithm was

proposed independently by Alon, Babai, and Itai [3]). Algorithm B of [18] (adapted to the message-passing model) operates in phases of three rounds, similarly to Algorithm A of [18] described in Section 2.1. In the first round, each active node v samples a random bit b that is 1 with probability $1/\max\{2d,1\}$, where d is the number of active neighbors of v, and broadcasts b to those neighbors along with d; if $b=1$ and for every active neighbor v' of v, its bit b' and number d' of its active neighbors satisfy $b'=0$ or $d'<d$, then v enters the MIS. The second and third rounds of a phase are identical to those of Algorithm A: newly added nodes to MIS inform their neighbors to remove themselves from the competition, and then removed nodes notify their neighbors. Unlike in Algorithm A, however, the third round is essential as it allows an active node v to update variable d to the current number of its active neighbors.

Algorithm 2 does not use phases. Also, instead of sending pair b,d, each active node broadcasts a single value $r=1+b\cdot d$; and each node with status *in* broadcasts the special value 0 similarly to Algorithm 1. The same `update()` procedure is used in both algorithms. An extra complication we have to deal with in the analysis of Algorithm 2 is that, at the beginning of round t, variable d does not contain the current number of active neighbors (i.e., the number at the end of round $t-1$), but rather the number at the end of round $t-2$.

We observe that Lemma 1 and Corollary 2 hold for Algorithm 2 as well, because they only rely on the `update()` procedure and on the premise that a node sends value 0 if and only if its status is *in*, and sends a positive value if and only if its status is *active*. From the algorithm it is also immediate that for any node v and round $t\geq 1$,

$$v.d_t = |N(v)\cap A_{t-1}|.$$

Next we bound the stabilization time of Algorithm 2. We start with a lemma which lower bounds the probability that at least one node from a given set $Z\subseteq A_t\setminus N(I_t)$ changes its status from *active* to *in* in round $t+1$. A similar statement was shown in [18, Lemma B].

Lemma 4. *Let $t\geq 3$, and let $Z\subseteq V$ be a non-empty set. If $Z\subseteq A_t\setminus N(I_t)$ then*

$$\mathbb{P}\left[Z\cap I_{t+1}\neq\emptyset \mid \mathcal{F}_t\right] \geq \frac{1}{4}\min\left\{1,\ \sum_{v\in Z}\frac{1}{\max\{1,2v.d_t\}}\right\}.$$

Proof. We fix the execution up to the end of round t, so we do not have to condition probabilities on $\mathcal{F}_t$, and suppose that $Z\subseteq A_t\setminus N(I_t)$. Let $Z=\{v_1,\dots,v_{|Z|}\}$, and for each $1\leq i\leq |Z|$, let $k_i=v_i.d_t=|N(v_i)\cap A_{t-1}|$.

We first argue that $k_i\neq 0$ for all $1\leq i\leq |Z|$: Since $v_i\in A_t$ we have also $v_i\in A_{t-1}$, because $A_t\subseteq A_{t-1}$ for $t\geq 3$ by Corollary 2. Suppose that $k_i=0$. Then $N(v_i)\cap A_{t-1}=\emptyset$ and thus $v.M_t$ is either $\emptyset$ or $\{0\}$. In either case, since $v_i\in A_{t-1}$ it follows from the `update()` procedure that $v_i\notin A_t$, which contradicts the definition of Z. Thus $k_i\neq 0$.

Let $\mathcal{E}$ be the event that $v.r_{t+1}>1$ for some $v\in Z$. Then

$$\begin{aligned}\mathbb{P}[\mathcal{E}] &= 1-\prod_{1\leq i\leq |Z|}\left(1-\frac{1}{2k_i}\right)\geq 1-e^{-\sum_{1\leq i\leq |Z|}\frac{1}{2k_i}}\\ &\geq \frac{1}{2}\min\left\{1,\ \sum_{1\leq i\leq |Z|}\frac{1}{2k_i}\right\},\end{aligned} \tag{1}$$

where in the first equation we used $v_i.d_t=k_i\neq 0$, and in the last inequality we used $e^{-x}\leq 1-x/2$ for $0\leq x\leq 1$.

For $1\leq i\leq |Z|$, let $\mathcal{E}_i$ be the event that $v_1.r_{t+1}=\cdots=v_{i-1}.r_{t+1}=1$ and $v_i.r_{t+1}>1$. Then $\mathcal{E}=\bigvee_{1\leq i\leq |Z|}\mathcal{E}_i$, and since the events $\mathcal{E}_i$ are disjoint, $\mathbb{P}[\mathcal{E}]=\sum_{1\leq i\leq |Z|}\mathbb{P}[\mathcal{E}_i]$.

We consider now the case in which event $\mathcal{E}_i$ occurs. For every $w\in N(v_i)\cap A_t$, either $w.r_{t+1}=1$ or $w.r_{t+1}=1+w.d_t$. Let $U_i=\{w\in N(v_i)\cap A_t:\ w.d_t\geq k_i\}$. We have that $v_i\in I_{t+1}$ if (and only if) $w.r_{t+1}=1$ for all $w\in U_i$: If $w.r_{t+1}=1$ for all $w\in U_i$ then $v_i.r_{t+1}=1+k_i>w.r_{t+1}$ for all $w\in N(v_i)\cap A_t$. And since $v_i\in A_t\setminus N(I_t)$, the last rule of `update()` applies, giving $v_i\in I_{t+1}$. Therefore,

$$\mathbb{P}[v_i\in I_{t+1}\mid\mathcal{E}_i]=\mathbb{P}\left[\bigwedge_{w\in U_i} w.r_{t+1}=1\ \middle|\ \mathcal{E}_i\right].$$

We have $|U_i|\leq |N(v_i)\cap A_t|\leq |N(v_i)\cap A_{t-1}|=v_i.d_t=k_i$, and conditioned on $\mathcal{E}_i$, for each $w\in U_i\setminus\{v_1,\dots,v_{i-1}\}$ the probability that $w.r_{t+1}=1$ is $1-\frac{1}{2w.d_t}\geq 1-\frac{1}{2k_i}$, while $v_j.r_{t+1}=1$ for $j<i$. It follows that

$$\mathbb{P}[v_i\in I_{t+1}\mid\mathcal{E}_i]\geq\left(1-\frac{1}{2k_i}\right)^{k_i}\geq 4^{-\frac{1}{2k_i}\cdot k_i}=\frac{1}{2},$$

where in the second inequality we used that $1-x\geq 4^{-x}$ when $0\leq x\leq 1/2$.

Finally, since the events $\mathcal{E}_i$ are disjoint,

$$\begin{aligned}\mathbb{P}[Z\cap I_{t+1}\neq\emptyset] &\geq \sum_{1\leq i\leq |Z|}\mathbb{P}[Z\cap I_{t+1}\neq\emptyset\mid\mathcal{E}_i]\cdot\mathbb{P}[\mathcal{E}_i]\\ &\geq \sum_{1\leq i\leq |Z|}\mathbb{P}[v_i\in I_{t+1}\mid\mathcal{E}_i]\cdot\mathbb{P}[\mathcal{E}_i]\\ &\geq \sum_{1\leq i\leq |Z|}\frac{1}{2}\cdot\mathbb{P}[\mathcal{E}_i]\\ &= \frac{1}{2}\cdot\mathbb{P}[\mathcal{E}].\end{aligned}$$

Combining this and (1) completes the proof. □

We will use the following definition of good nodes and good edges from [3], and also the fact that at least half of the edges of a graph are good.

Definition 5 *(Good Nodes & Edges).* Let $H=(V_H,E_H)$ be a graph. A node $v\in V_H$ of H is *good* if

$$\left|\{u\in N_H(v):\ \deg_H(u)\leq\deg_H(v)\}\right|>\deg_H(v)/3,$$

i.e., the degree of more than $1/3$ of v's neighbors is smaller than or equal to the degree of v. An edge $\{v,u\}\in E_H$ is *good* if at least one of v and u is a good node.

Lemma 6 *([3, Lemma 4.4]). At least half of the edges in any graph are good edges.*

Recall that for any set $X\subseteq V$, $G[X]=(X,E(X))$ is the induced subgraph of G on X. The next technical lemma compares the graphs $G[A_{t-1}\setminus I_t]$ and $G[A_{t+1}\setminus O_{t+2}]=G[A_{t+1}\setminus N(I_{t+1})]$. It shows that for any good node v in $G[A_{t-1}\setminus I_t]$, at least a constant expected fraction of its incident edges no longer exist in $G[A_{t+1}\setminus O_{t+2}]$. The proof proceeds by applying Lemma 4 to an appropriate subset of the neighbors of good node v.

Lemma 7. *Let $t\geq 3$, and for any $s\geq 1$ let*

$$X_s=A_{s-1}\setminus I_s,\quad Y_s=A_s\setminus N(I_s).$$

If v is a good node of $G[X_t]$, the expected number of its incident edges $\{v,u\}\in E(X_t)$ not belonging to $E(Y_{t+1})$ is

$$\mathbb{E}\left[\left|E(v,X_t)\setminus E(Y_{t+1})\right|\mid\mathcal{F}_t\right]\geq\left|E(v,X_t)\right|/30.$$

Proof. Let $k=\left|E(v,X_t)\right|=\left|N(v)\cap X_t\right|$ be the number of neighbors of v in $G[X_t]$. The statement is obviously true if $k=0$, so we assume $k\neq 0$.

We first consider the simple case in which at least $k/30$ of v's neighbors in $G[X_t]$ belong to $N(I_t)$, i.e.,

$$\left|N(v)\cap X_t\cap N(I_t)\right|\geq k/30.$$

In this case, it suffices to consider just the neighbors of v in $N(I_t)$ to establish the claim. Precisely, since $N(I_t)\cap Y_{t+1} \subseteq N(I_t)\cap A_{t+1} = O_{t+1}\cap A_{t+1} = \emptyset$, we have

$$\left|E(v,X_t)\setminus E(Y_{t+1})\right| \geq \left|N(v)\cap X_t \cap N(I_t)\right| \geq k/30,$$

which implies the lemma.

We consider now the case where $\left|N(v)\cap X_t\cap N(I_t)\right| < k/30$. In this case, we apply Lemma 4 to the following subset Z of v's neighbors in $G[X_t]$,

$$Z = \{u\in (N(v)\cap X_t)\setminus N(I_t) :\ \left|N(u)\cap X_t\right| \leq k\}.$$

Before doing so, we prove that condition $Z\subseteq A_t\setminus N(I_t)$ of Lemma 4 is met, and also show that $|Z|\geq 3k/10$, and $u.d_t \leq k$, for $u\in Z$.

Since v is a good node in $G[X_t]$, at least $k/3$ of v's neighbors in $G[X_t]$ have degree at most k. Among these, at least $k/3 - k/30 = 3k/10$ are not in $N(I_t)$, by the case hypothesis. Therefore,

$$|Z| \geq 3k/10.$$

Also

$$Z\subseteq A_t\setminus N(I_t),$$

because $Z\subseteq X_t\setminus N(I_t)$ and

$$\begin{aligned}(X_t\setminus N(I_t))\setminus (A_t\setminus N(I_t)) &= (X_t\setminus A_t)\setminus N(I_t)\\ &= (A_{t-1}\setminus (I_t\cup A_t))\setminus N(I_t)\subseteq O_t\setminus N(I_t)=\emptyset,\end{aligned}$$

where in the last equation we used $N(I_t) = O_{t+1}\supseteq O_t$, by Corollary 2. Finally, we have that if $u\in Z$, then $u\notin N(I_t)$, which implies $N(u)\cap I_t = \emptyset$, and thus $N(u)\cap A_{t-1} = N(u)\cap (A_{t-1}\setminus I_t) = N(u)\cap X_t$. Hence, if $u\in Z$,

$$u.d_t = |N(u)\cap A_{t-1}| = |N(u)\cap X_t| \leq k,$$

by the definition of Z.

We can now apply Lemma 4 to obtain

$$\mathbb{P}[Z\cap I_{t+1}\neq\emptyset \mid \mathcal{F}_t] \geq \frac{1}{4}\min\left\{1, \frac{3k}{10}\cdot\frac{1}{2k}\right\} = \frac{3}{80} > \frac{1}{30}.$$

If $Z\cap I_{t+1}\neq\emptyset$, then $v\in N(I_{t+1})$, and thus $v\notin Y_{t+1}$ and $\left|E(v,X_t)\setminus E(Y_{t+1})\right| = E(v,X_t)$. It follows that

$$\mathbb{E}\left[\left|E(v,X_t)\setminus E(Y_{t+1})\right| \mid \mathcal{F}_t\right] \geq \frac{1}{30}\cdot\left|E(v,X_t)\right|,$$

as desired. □

We are now ready to prove the main statement.

Theorem 8. *Algorithm 2 computes an MIS in $O(\log n)$ rounds w.h.p.*

Proof. Define $X_t = A_{t-1}\setminus I_t$ and $Y_t = A_t\setminus N(I_t)$, as in Lemma 7, and let $X_t^g\subseteq X_t$ and $E^g(X_t)\subseteq E(X_t)$ be the set of good nodes and the set of good edges of graph $G[X_t]$, respectively. Let $t\geq 3$. The expected number of edges in $E(X_t)$ not belonging to $E(Y_{t+1})$ is

$$\begin{aligned}\mathbb{E}[\left|E(X_t)\setminus E(Y_{t+1})\right| \mid \mathcal{F}_t] &= \frac{1}{2}\sum_{v\in X_t}\mathbb{E}[\left|E(v,X_t)\setminus E(Y_{t+1})\right| \mid \mathcal{F}_t]\\ &\geq \frac{1}{2}\sum_{v\in X_t^g}\mathbb{E}[\left|E(v,X_t)\setminus E(Y_{t+1})\right| \mid \mathcal{F}_t]\\ &\geq \sum_{v\in X_t^g}\left|E(v,X_t)\right|/60\\ &\geq \left|E^g(X_t)\right|/120\\ &\geq \left|E(X_t)\right|/240,\end{aligned}$$

where in the third line we used Lemma 7, and in the last we used Lemma 6. It follows that

$$\mathbb{E}[\left|E(X_t)\setminus E(Y_{t+1})\right|] \geq \mathbb{E}[\left|E(X_t)\right|]/240.$$

From this and $E(Y_{t+1})\subseteq E(X_t)$, which holds because $Y_{t+1}\subseteq A_{t+1}\subseteq A_t = A_t\setminus I_t\subseteq A_{t-1}\setminus I_t = X_t$, we obtain

$$\mathbb{E}[\left|E(Y_{t+1})\right|] \leq (239/240)\cdot\mathbb{E}[\left|E(X_t)\right|].$$

This further implies

$$\mathbb{E}[\left|E(A_{t+2})\right|] \leq (239/240)\cdot\mathbb{E}[\left|E(A_{t-1})\right|],$$

since $Y_{t+1}\supseteq A_{t+2}\setminus N(I_{t+1}) = A_{t+2}\setminus O_{t+2} = A_{t+2}$, and $X_t\subseteq A_{t-1}$. Finally, applying repeatedly the last inequality above and using Markov's inequality, similarly to the proof of Theorem 3, we obtain that $|A_t| = 0$ w.h.p. for $t = \Theta(\log n)$ large enough; and the theorem follows by applying Corollary 2. □

CRediT authorship contribution statement

George Giakkoupis: Formal analysis. **Volker Turau:** Formal analysis. **Isabella Ziccardi:** Formal analysis.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgements

Support is gratefully acknowledged from the Agence Nationale de la Recherche (ANR) under project ByBloS (ANR-20-CE25-0002), and the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement No. 834861).

References

[1] Y. Afek, N. Alon, Z. Bar-Joseph, A. Cornejo, B. Haeupler, F. Kuhn, Beeping a maximal independent set, Distrib. Comput. 26 (4) (2013) 195–208, https://doi.org/10.1007/s00446-012-0175-7.
[2] Y. Afek, S. Dolev, Local stabilizer, J. Parallel Distrib. Comput. 62 (5) (2002) 745–765, https://doi.org/10.1006/JPDC.2001.1823.
[3] N. Alon, L. Babai, A. Itai, A fast and simple randomized parallel algorithm for the maximal independent set problem, J. Algorithms 7 (4) (1986) 567–583, https://doi.org/10.1016/0196-6774(86)90019-2.
[4] B. Awerbuch, S. Kutten, Y. Mansour, B. Patt-Shamir, G. Varghese, Time optimal self-stabilizing synchronization, in: Proc. 25th Annual ACM Symp. on Theory of Computing, STOC, 1993, pp. 652–661.
[5] A. Balliu, S. Brandt, J. Hirvonen, D. Olivetti, M. Rabie, J. Suomela, Lower bounds for maximal matchings and maximal independent sets, J. ACM 68 (5) (2021) 39:1–39:30, https://doi.org/10.1145/3461458.
[6] L. Barenboim, M. Elkin, U. Goldenberg, Locally-iterative distributed (Δ + 1)-coloring and applications, J. ACM 69 (1) (2022) 5:1–5:26, https://doi.org/10.1145/3486625.
[7] L. Barenboim, M. Elkin, S. Pettie, J. Schneider, The locality of distributed symmetry breaking, J. ACM 63 (3) (2016) 20:1–20:45, https://doi.org/10.1145/2903137.
[8] A. Cornejo, F. Kuhn, Deploying wireless networks with beeps, in: Proc. Int. Symp. on Distributed Computing, DISC, 2010, pp. 148–162.
[9] E.W. Dijkstra, Self-stabilizing systems in spite of distributed control, Commun. ACM 17 (11) (1974) 643–644, https://doi.org/10.1145/361179.361202.
[10] S. Dolev, Self-Stabilization, MIT Press, 2000.
[11] Y. Emek, E. Keren, A thin self-stabilizing asynchronous unison algorithm with applications to fault tolerant biological networks, in: Proc. ACM Symp. on Principles of Distributed Computing, PODC, 2021, pp. 93–102.
[12] M. Ghaffari, C. Grunau, V. Rozhon, Improved deterministic network decomposition, in: Proc. ACM-SIAM Symp. on Discrete Algorithms, SODA, 2021, pp. 2904–2923.

[13] G. Giakkoupis, I. Ziccardi, Distributed self-stabilizing MIS with few states and weak communication, in: Proc. ACM Symp. on Principles of Distributed Computing, PODC, 2023, pp. 310–320.
[14] W. Goddard, S.T. Hedetniemi, D.P. Jacobs, P.K. Srimani, Self-stabilizing protocols for maximal matching and maximal independent sets for ad hoc networks, in: Proc. Int. Parallel and Distributed Processing Symp., IPDPS, 2003, p. 162.
[15] N. Guellati, H. Kheddouci, A survey on self-stabilizing algorithms for independence, domination, coloring, and matching in graphs, J. Parallel Distrib. Comput. 70 (4) (2010) 406–415, https://doi.org/10.1016/j.jpdc.2009.11.006.
[16] M. Ikeda, S. Kamei, H. Kakugawa, A space-optimal self-stabilizing algorithm for the maximal independent set problem, in: Proc. Int. Conf. on Parallel and Distributed Computing, Applications and Technologies, PDCAT, 2002, pp. 70–74.
[17] C. Lenzen, J. Suomela, R. Wattenhofer, Local algorithms: self-stabilization on speed, in: Proc. Int. Symp. on Stabilization, Safety, and Security of Distributed Systems, SSS, 2009, pp. 17–34.
[18] M. Luby, A simple parallel algorithm for the maximal independent set problem, SIAM J. Comput. 15 (4) (1986) 1036–1053, https://doi.org/10.1137/0215074.
[19] N.A. Lynch, Distributed Algorithms, Morgan Kaufmann, 1996.
[20] Y. Métivier, J.M. Robson, N. Saheb-Djahromi, A. Zemmari, An optimal bit complexity randomized distributed MIS algorithm, Distrib. Comput. 23 (5–6) (2011) 331–340, https://doi.org/10.1007/s00446-010-0121-5.
[21] S.K. Shukla, D.J. Rosenkrantz, S.S. Ravi, Observations on self-stabilizing graph algorithms for anonymous networks, in: Proc. Workshop on Self-Stabilizing Systems, SSS, 1995.
[22] V. Turau, Linear self-stabilizing algorithms for the independent and dominating set problems using an unfair distributed scheduler, Inf. Process. Lett. 103 (3) (2007) 88–93, https://doi.org/10.1016/j.ipl.2007.02.013.