Self-stabilizing Cuts in Synchronous Networks

Thomas Sauerwald[1,⋆] and Dirk Sudholt[2,⋆⋆]

[1] Dept. of CS, University of Paderborn, Paderborn, Germany
sauerwal@upb.de
[2] Dept. of CS, Dortmund University of Technology, Dortmund, Germany
dirk.sudholt@cs.uni-dortmund.de

Abstract. Consider a synchronized distributed system where each node can only observe the state of its neighbors. Such a system is called self-stabilizing if it reaches a stable global state in a finite number of rounds. Allowing two different states for each node induces a cut in the network graph. In each round, every node decides whether it is (locally) satisfied with the current cut. Afterwards all unsatisfied nodes change sides independently with a fixed probability p. Using different notions of satisfaction enables the computation of maximal and minimal cuts, respectively. We analyze the expected time until such cuts are reached on several graph classes and consider the impact of the parameter p and the initial cut.

1 Introduction

1.1 Motivation

In the language of distributed computing a system is called self-stabilizing if it reaches a global state with some desired property in finite time, regardless of the initialization. This implies that the system is able to stabilize even in the presence of faults [2,4]. Such self-stabilizing processes have been investigated for various graph problems like maximal matchings [11,15], independent sets [8], and domination [6]. A lot of research effort has been spent on self-stabilizing vertex coloring algorithms [7,9,12,13,14], motivated by code assignment problems in wireless networks.

In this work we consider self-stabilizing algorithms for maximal and minimal cuts in a synchronized distributed system. The network is given by an undirected graph $G = (V, E)$. As we do not make use of IDs for the nodes, we assume that the network is anonymous. However, we assume that there is a central clock synchronization. In each round every node has one out of two possible states, which induces a cut of the network. In every round every node decides whether it is satisfied with the current cut, judging from a local perspective, i. e., the state

⋆ Supported by the German Science Foundation (DFG) Research Training Group GK-693 of the Paderborn Institute for Scientific Computation (PaSCo).
⋆⋆ Supported by the German Science Foundation (DFG) as a part of the Collaborative Research Center "Computational Intelligence" (SFB 531).

A. Shvartsman and P. Felber (Eds.): SIROCCO 2008, LNCS 5058, pp. 234–246, 2008.

of its neighbors. Unsatisfied nodes strive to (locally) improve the cut by changing sides. In order to break symmetries, we investigate a randomized algorithm where in each round every unsatisfied node changes sides with a fixed probability p.

By different notions of satisfaction different types of cuts can be produced. We say that a node is *max-satisfied* if at least half of its neighbors are on the other side of the cut. If all nodes are max-satisfied, the current cut cannot be increased by flipping a single node. Hence the current cut is *maximal*, i. e., locally optimal w. r. t. the cut size (as opposed to *maximum* cuts representing global optima). From a global perspective, the system may be viewed as a self-stabilizing algorithm for maximal cuts.

The system may also be regarded from a local perspective. For example, the problem can be seen as a relaxed code assignment problem where nodes are forced to use different codes to communicate. In a cut where all nodes are max-satisfied every node can communicate with a majority of neighbors, even if only two codes are available. There are also connections to game theory where the nodes represent players competing for services. If some players asking for the same service are close to each other (are connected by an edge), then the benefit of this service has to be split among all these players.

On the other hand, a node is *min-satisfied* if at least half of its neighbors are on the same side of the cut. This notion of satisfaction results in *minimal* cuts (as opposed to *minimum* cuts). Finding a minimum cut in a graph is an important task in computer science with applications to clustering, chip design, and network reliability. In our distributed and anonymous setting, however, we are content with minimal cuts.

Using the above-mentioned two notions of satisfaction, we show that the system self-stabilizes and then focus on the expected time until a stable cut is obtained. We prove for both satisfaction models that planar graphs stabilize in linear time for appropriate constant values of p. The choice of p is crucial since using constant p on dense graphs results in exponential stabilization times for the max-satisfaction model, with high probability. Finally, we investigate classes of sparse graphs like rings, torus graphs, and hypercubes. On rings the expected stabilization time is logarithmic for constant p. For some torus graphs, the choice of the initial cut decides between linear and logarithmic expected stabilization times.

1.2 Related Work

Our work is related to the design of distributed approximation algorithms [5] since our algorithm approximates maximum and minimum cuts. This is especially interesting as Elkin [5] concludes in his survey that the distributed approximability of maximum and minimum cut is still unsolved. However, the focus on this work is different; due to the restrictions in our distributed model we only settle for maximal and minimal cuts, i. e., local optima.

Gradinariu and Tixeuil [9] investigated a self-stabilizing coloring algorithms that is similar to our model. In their work, a node agrees with its neighborhood if it is colored with the maximal color value that is not used by any of its neighbors.

In their distributed setting a node that disagrees with its neighborhood changes its color with probability 1/2. It is shown that this strategy stabilizes with a $(B+1)$-coloring in expected time $O((B-1)\log n)$ where B is a bound on the maximal degree and n is the number of nodes. This work loosely relates to our work as every 2-coloring represents a maximum cut. However, as typically $B+1 > 2$ colors may be used, vertex coloring and cut problems are quite different.

1.3 Our Results

After presenting necessary definitions in Section 2, we start with general upper bounds for the expected stabilization time in both min-satisfaction and max-satisfaction models in Section 3. In particular, we derive an upper bound $O(n/p)$ for all planar graphs with n nodes if $p \leq 1/12$. This bound suggests to choose p large, but for dense graphs this may lead to exponential stabilization times. Section 4 presents such examples for the max-satisfaction process on the complete graph K_n and dense random graphs in the $\mathcal{G}(n,1/2)$-model. On K_n the expected stabilization time is exponential for $p = 1/2$, but polynomial if $p = O((\log n)/n)$ (and $p \geq n^{-O(1)}$). For sparse graphs the choice of p is less important. As shown in Section 5, rings stabilize in expected time $O((\log n)/p)$ if $p = 1 - \Omega(1)$. Moreover, the investigation of torus graphs shows that the initialization can be crucial. With a worst-case initialization torus graphs stabilize in expected time $\Omega(n/p)$, while random initialization yields a bound of $O((\log n)/p^2)$ on certain torus graphs. Section 6 finishes with conclusions and remarks on future work. Due to space limitations proofs from Section 4 are omitted. An extended version with these proofs is available as technical report [16].

2 Definitions

Let $G = (V, E)$ be an undirected graph. For $U, W \subseteq V$ let $E(U, W)$ be the set of all edges between U and W and $E(U) = E(U, U)$. For $v \in V$ let $\deg(v)$ denote the degree of v. Let $\Delta(G) = \max_{v \in V} \deg(v)$ be the maximum degree in G and $a(G) = \max_{U \subseteq V, |U|>1} \left\lceil \frac{|E(U)|}{|U|-1} \right\rceil$ be the (edge) arboricity of G (see [1]). We use $a(G)$ as a measure of local density in the graph and observe that $a(G)$ is small iff G is "nowhere dense." The number of nodes is always denoted by n.

At each point of time all nodes are either in state 0 or in state 1. In round t let $V_t(1) \subseteq V$ denote the set of nodes in state 1; $V_t(0) = V \backslash V_t(1)$ is the corresponding complementing set. We synonymously use the term coloring and say that a node v is c-colored if $v \in V_t(c)$, $c \in \{0, 1\}$. In this case we denote $\deg_t^+(v) = |E(\{v\}, V_t(1-c))|$ and $\deg_t^-(v) = \deg(v) - \deg_t^+(v)$. We define two notions of satisfaction mentioned before.

Definition 1. *A node v is max-satisfied at time t if $\deg_t^+(v) \geq \deg_t^-(v)$. A node v is min-satisfied at time t if $\deg_t^+(v) \leq \deg_t^-(v)$.*

Fixing one notion of satisfaction, let V_t^{sat} denote the set of all nodes that are satisfied at time t and $V_t^{\text{unsat}} := V \setminus V_t^{\text{sat}}$ denote the set of unsatisfied nodes. Given $0 < p < 1$, the self-stabilizing cut algorithm is formally defined as follows.

SELF-STABILIZING CUT ALGORITHM

1: In round t execute the following rule simultaneously for all nodes v:
2: if $v \in V_t^{\text{unsat}}$ then
3: invert state of v for round $t+1$ with probability p.

A cut where all nodes are satisfied is called *stable*. The stabilization time is defined as the first round with a stable cut. We are interested in the expected stabilization time, where the initial cut may be chosen uniformly at random or by an adversary. In the latter case, we speak of the worst-case expected stabilization time.

Observe that for bipartite graphs one can easily switch between the two models of satisfaction. Given a bipartition $V = U \cup W$ of the graph $G = (V, E)$, flipping (inverting) all nodes in U turns every cut edge into a non-cut edge and vice versa. Thereby, the meaning of $\deg_t^+(v)$ and $\deg_t^-(v)$ is exchanged and a node becomes min-satisfied iff it has been max-satisfied before. In particular, a stable cut for one model becomes a stable cut for the other model after this transformation.

More precise, let the function h on the state space $\{0,1\}^n$ be such a transformation, then the following holds. Consider the algorithm applied to both models. If the max-satisfaction model starts in state x_0 and the min-satisfaction model starts in state $y_0 = h(x_0)$, then at any point of time t for any state x_t the probability that the max-satisfaction model is in state x_t equals the probability that the min-satisfaction model is in state $y_t = h(x_t)$. This symmetrical behavior implies that the random stabilization times for the two models have the same probability distribution. It therefore suffices to focus on one model when dealing with bipartite graphs.

In the max-satisfaction model, shortly max-model, a stable configuration represents a maximal cut, i. e., a cut that cannot be enlarged by changing a single node. This is because a local improvement implies an unsatisfied node. The same holds for the min-model and minimal cuts. In a non-distributed setting one may easily obtain maximal and minimal cuts by local search, simply changing a single unsatisfied node in each round. The number of cut edges is then strictly increasing over time, implying that at most $|E|$ iterations are needed in order to find a maximal or minimal cut. The self-stabilizing cut algorithm can simulate an iteration of local search if exactly one specific unsatisfied node is flipped, which happens with probability $p \cdot (1-p)^{|V_t^{\text{unsat}}|-1} > 0$. Hence, there is a positive probability that the algorithm simulates a whole run of local search ending with a stable cut.

Proposition 1. *In both the max-model and the min-model, the self-stabilizing cut algorithm stabilizes in finite time with probability 1.*

In the following, we will present more precise results, i. e., we prove bounds between logarithmic, polynomial, and exponential orders for different graph classes. As we are especially interested in the impact of the parameter p, we state our results w. r. t. n and p.

3 A General Upper Bound

In this section we derive general upper bounds for both the max-model and the min-model. Thereby, we exploit that under certain conditions there is a probabilistic tendency to increase the cut size in the max-model and to decrease the cut size in the min-model, respectively.

Theorem 1. *On any graph $G = (V, E)$, if $p \leq 1/(4a(G))$, the expected stabilization time for both the max-model and the min-model is bounded from above by $2|E|/p$.*

Proof. It suffices to consider the max-model as the arguments for the min-model are symmetric. Let $\mathcal{P}_t = (V_t(0), V_t(1))$ and let $f(\mathcal{P}_t)$ be the number of cut edges in $\mathcal{P}_t$. Consider one round of the algorithm and let V_t^{flip} be the set of nodes changing sides (flipping) in round t. If v is the only node to be flipped in round t, this operation increases the cut size by $\deg_t^-(v) - \deg_t^+(v) \geq 1$. If V_t^{flip} is an independent set, the total increase of the cut size is $\sum_{v \in V_t^{\text{flip}}} (\deg_t^-(v) - \deg_t^+(v)) \geq |V_t^{\text{flip}}|$. However, if two changing nodes share an edge, this edge is counted wrongly for both nodes. This implies

$$\begin{aligned} f(\mathcal{P}_{t+1}) - f(\mathcal{P}_t) &\geq \sum_{v \in V_t^{\text{flip}}} (\deg_t^-(v) - \deg_t^+(v)) - 2|E(V_t^{\text{flip}})| \\ &\geq |V_t^{\text{flip}}| - 2|E(V_t^{\text{flip}})|. \end{aligned}$$

The expected gain in cut size is at least

$$\mathbf{E}\left(f(\mathcal{P}_{t+1}) - f(\mathcal{P}_t)\right) \geq p|V_t^{\text{unsat}}| - 2p^2|E(V_t^{\text{unsat}})|.$$

Observe $|E(V_t^{\text{unsat}})| \leq a(G) \cdot (|V_t^{\text{unsat}}| - 1) < a(G) \cdot |V_t^{\text{unsat}}|$ by definition of $a(G)$. Along with the assumption $p \leq 1/(4a(G))$, we arrive at

$$\mathbf{E}\left(f(\mathcal{P}_{t+1}) - f(\mathcal{P}_t)\right) \geq p|V_t^{\text{unsat}}| - 2p^2 \cdot a(G) \cdot |V_t^{\text{unsat}}| \geq p/2 \cdot |V_t^{\text{unsat}}|.$$

As long as the current cut is not stable, $|V_t^{\text{unsat}}| \geq 1$, hence the expected increase in cut size is at least $p/2$.

We now use drift analysis arguments from He and Yao [10, Lemma 1]. Consider a Markov chain with states $X_0, X_1, \ldots$ for domain $\mathbb{R}_0^+$. Let $\alpha, \delta > 0$ and assume we are interested in the first time until the Markov chain first reaches a value at least α. If δ is a lower bound for the expected increase in one step, i. e., $\mathbf{E}\left(X_{t+1} - X_t \mid X_t\right) \geq \delta$ for $X_t < \alpha$, the expected first hitting time for a value at least α is at most α/δ. Symmetrically, if $\mathbf{E}\left(X_t - X_{t+1} \mid X_t\right) \geq \delta$ for $X_t > 0$, the expected time to reach value 0 starting with α is at most α/δ.

We apply these statements to the random cut size and finish our considerations prematurely if a maximal cut is reached. Hence, the expected time until a cut of size $|E|$ is reached or a maximal cut is found beforehand is bounded by $|E|/(p/2) = 2|E|/p$. □

Section 5 contains examples where this bound is asymptotically tight. Note that the simple strategy of choosing $p = 1/(2n)$ is oblivious of the graph at hand and, nevertheless, yields a polynomial bound of $4|E|n$ rounds. This also proves that the expected stabilization time can be polynomial for *any* graph if the parameter p is chosen appropriately.

From Theorem 1 one can easily derive a handy upper bound for all planar graphs. The arboricity of a planar graph is known to be at most 3. A proof follows by contradiction. If there is a set $U \subseteq V$ with $|U| > 1$ such that $a(G) \geq \frac{|E(U)|}{|U|-1} > 3$, this implies $|E(U)| > 3|U| - 3$. However, this contradicts the fact that the number of edges in a planar graph with k nodes is at most $3k - 6$ (see, e. g., [3]). Therefore $a(G) \leq 3$ holds if G is planar.

Corollary 1. *On any planar graph $G = (V, E)$, if $p \leq 1/12$, the expected stabilization time for the max-model and the min-model is bounded by $2|E|/p \leq 6n/p$.*

4 Dense Graphs

The upper bounds from the previous section grow with $1/p$, suggesting to always choose p large. In this section, however, we prove for the max-model that in dense graphs large values for p may result in exponentially large stabilization times. The complete graph K_n is the simplest dense graph. For even n, a cut is maximal (and maximum in this case) if $|V_t(0)| = n/2$. However, if p is chosen too large, it may happen that too many nodes change sides simultaneously and a majority of 0-nodes is turned into a similarly large majority of 1-nodes, and so forth. This may result in a non-stable equilibrium that is hard to overcome. The following result shows that for large p the max-model needs exponential time to stabilize. Due to space limitations, proofs for the following theorems are placed in an extended version of this work [16].

Theorem 2. *Consider the complete graph K_n, n even, with $n^{-1/3} \leq p \leq 1/2$ and an arbitrary, non-stable initialization. Then the stabilization time of the max-model is at least $\frac{1}{2}\exp(\frac{np^3}{192})$ with probability $1 - o(1)$.*

On the other hand, the effect of too many flipping nodes decreases with decreasing p. The following result shows that if $p = O((\log n)/n)$ (and, of course, $p \geq n^{-O(1)}$) the expected stabilization time is polynomial.

Theorem 3. *Consider the complete graph K_n, n even, with an arbitrary initialization. Then the expected stabilization time of the max-model is bounded above by $1/p \cdot (1-p)^{-n/2}$.*

Negative results for an unlucky initialization can also be shown for random graphs of a probability space $\mathcal{G}(n,p')$ defined as follows. The random graph consists of n nodes and between any pair of nodes, an edge occurs independently with probability p'. The case $p'=1/2$ is especially interesting as $G \in \mathcal{G}(n,1/2)$ is a uniform sample among all graphs with n nodes.

Theorem 4. *Consider a graph G in $\mathcal{G}(n,1/2)$, n even, and assume that initially $\frac{20}{32}n \leq |V_0(0)| \leq \frac{23}{32}n$. Then the stabilization time of the max-model with $p=\frac{1}{2}$ is $\exp(\Omega(n))$ with probability $1-\exp(-\Omega(n))$ (w. r. t. the randomized construction of G and the randomized self-stabilizing cut algorithm).*

5 Ring Graphs, Torus Graphs, and Hypercubes

We now consider commonly used network topologies like ring graphs (and other graphs with maximum degree 2), torus graphs, and hypercubes.

5.1 Ring Graphs

Consider a graph $G=(V,E)$ with maximum degree 2. Theorem 1 yields an upper bound $O(n/p)$ if $p \leq 1/8$. We improve upon this result exploiting that on these topologies satisfied nodes cannot become unsatisfied again.

Definition 2. *A set of nodes $S \subseteq V$ is called* stable *w. r. t. the current cut $\mathcal{P}_t$ if all nodes in S are satisfied and will remain so in all future rounds with probability 1. A node v is called stable if it is contained in a stable set; otherwise, v is called unstable.*

Isolated nodes are trivially stable, hence we assume that G does not contain isolated nodes. Then in the max-model (min-model) a node v is satisfied iff it has at least one neighbor w on the other side of the cut (on the same side of the cut). This condition also implies that w is satisfied. Even stronger, v and w will remain satisfied forever since the edge $\{v,w\}$ will never be touched again. Therefore, on graphs with maximum degree 2 all satisfied nodes are stable.

Theorem 5. *The expected stabilization time for the max-model and the min-model on a graph $G=(V,E)$ with $\Delta(G) \leq 2$ is $O((\log n)/(p(1-p)))$.*

Proof. Consider a node v that is unsatisfied in round t and the random decision whether to flip v or not. At least one decision makes v satisfied in round $t+1$. The "right" random decision for v is made with probability at least $q := \min\{p, 1-p\}$. In expectation $q|V_t^{\text{unsat}}|$ nodes become satisfied (and therefore stable), hence $\mathbf{E}\left(|V_{t+1}^{\text{unsat}}| \mid |V_t^{\text{unsat}}|\right) \leq (1-q)\cdot|V_t^{\text{unsat}}|$ for any $V_t^{\text{unsat}} \subseteq V$. Using the law $\mathbf{E}\left(|V_{t+1}^{\text{unsat}}|\right) = \mathbf{E}\left(\mathbf{E}\left(|V_{t+1}^{\text{unsat}}|\right) \mid |V_t^{\text{unsat}}|\right)$ and a trivial induction yields $\mathbf{E}\left(|V_t^{\text{unsat}}|\right) \leq (1-q)^t \cdot |V_0^{\text{unsat}}| \leq (1-q)^t \cdot n$.

Choosing $T := \left\lceil \log_{(1-q)} \frac{1}{2n} \right\rceil$ yields $\mathbf{E}\left(|V_T^{\text{unsat}}|\right) \leq 1/2$. By Markov's inequality $\mathbf{Pr}\left(|V_T^{\text{unsat}}| \geq 1\right) \leq 1/2$. Hence after T rounds all nodes are satisfied with

probability at least 1/2, regardless of the initial cut. If this is not the case, we consider another period of T rounds and repeat the argumentation. The expected number of periods is at most 2, hence the expected stabilization time is bounded by

$$2T \leq 2\left(\log_{(1-q)} \frac{1}{2n}\right) + 2 = \frac{2\ln(2n)}{\ln\left(\frac{1}{1-q}\right)} + 2 \leq \frac{2\ln(2n)}{q} + 2 = O\left(\frac{\log n}{q}\right)$$

where the second inequality follows from $1/(1-x) \geq e^x$ for $x < 1$. The theorem follows since $q = \Theta(p(1-p))$. □

5.2 Torus Graphs

We denote by $G_{r\times s} = (V, E)$ for $r, s \geq 4$ both even a two-dimensional torus graph, defined by

$$\begin{aligned} V &= \{(x,y) \mid 0 \leq x \leq r-1, 0 \leq y \leq s-1\} \text{ and} \\ E &= \{(x_1,y_1),(x_2,y_2) \mid (x_2 = x_1 \wedge y_2 = (y_1+1) \bmod s) \vee \\ &\qquad (x_2 = (x_1+1) \bmod r \wedge y_2 = y_1)\}. \end{aligned}$$

$G_{r\times s}$ thus consists of r rows and s columns (see Figure 1). Note that due to the assumptions on r and s all torus graphs are bipartite and regular as all nodes have degree 4. Recall that the max-model can be transferred into an equivalent min-model by inverting states of all nodes in one set of the bipartition. The visualization is easier for the min-model where large monochromatic areas in the torus are "good." Hence we will argue with the min-model in the following; however, all results also hold for the max-model.

In the min-model we can derive an intuitive characterization of stable nodes, referring to states synonymously as colors. A sufficient condition for a c-colored node v to be stable is that v belongs to a cycle of c-colored nodes. Moreover, v is stable if it belongs to a path connecting two such cycles. The following lemma shows that these two conditions are also necessary for stability.

Lemma 1. *Consider the min-model for $G_{r\times s}$. A c-colored node v, $c \in \{0,1\}$, is stable iff v belongs to a cycle of c-colored nodes or v is on a path of c-colored nodes connecting two such cycles.*

Proof. Consider the subgraph $G_c = (V_c, E_c)$ induced by all c-colored nodes. On a cycle $C \subseteq V_c$ all $u \in C$ are satisfied, hence they will remain so forever. Consider a path $P \subseteq V_c$ connecting two cycles $C_1, C_2 \subseteq V_c$. As all nodes in $C_1 \cup C_2$ remain satisfied, all $u \in P$ remain satisfied as well.

On the other hand, if v is neither on a cycle nor on a path connecting two cycles, then v cannot be stable. Assume that v is satisfied since otherwise the claim is trivial. Let S be the union of all cycles in V_c, then $v \in V_c \setminus S$. Let T be the connected component of $V_c \setminus S$ that contains v. As T does not contain cycles, T is a tree. Consider v as the root of T, then v has at least two subtrees in T

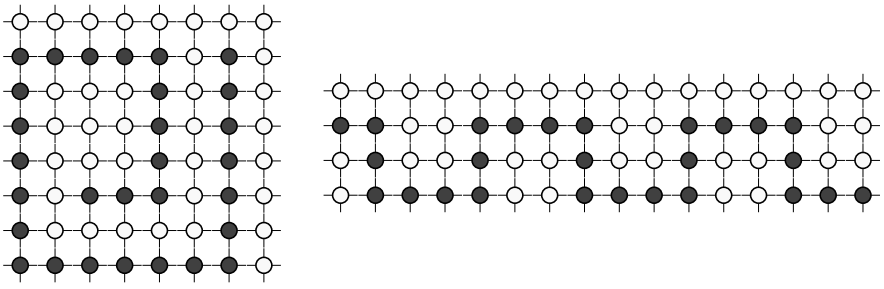

Fig. 1. Torus graphs $G_{8\times 8}$ (left) and $G_{4\times 16}$ (right). The coloring shows worst-case initial cuts in the min-model, where only the end nodes of the black paths are unsatisfied. All white nodes are stable by Lemma 1.

since v is satisfied. As v does not lie on a path connecting two cycles, at most one of v's subtrees is connected to S. In a subtree not connected to S every leaf is unsatisfied. If the next subsequent rounds only flip leaves of T, all subtrees of v (except one, if v is connected to S) are gradually eliminated, leaving v unsatisfied. We conclude that v cannot be stable. □

We first consider the worst-case expected stabilization time. It is easy to see that we can color the nodes in $G_{r\times s}$ such that all 1-nodes form a path of length $\Omega(n)$ where every 1-node is adjacent to at most two other 1-nodes and all 0-nodes are stable. Figure 1 gives two examples. In such a cut only the two ends of the path are unsatisfied. As long as the path has length at least 2, this property is preserved since flipping an end node renders its neighbor on the path unsatisfied. The algorithm is thus forced to flip the nodes on this path one after another, starting from both ends simultaneously. It is then easy to prove the following lower bound.

Theorem 6. *The worst-case expected stabilization time for both the max-model and the min-model on $G_{r\times s}$ is $\Omega(n/p)$.*

An upper bound can be shown using that unsatisfied nodes have a good chance to become part of a cycle of equally colored nodes.

Lemma 2. *Consider the torus graph $G_{r\times s}$. If the current cut contains an unsatisfied node v, the probability that v becomes stable within the next two rounds is at least $p^2(1-p)^5$.*

Proof. W. l. o. g. v is 1-colored and we consider the min-model. We name nodes around v according to their direction from v and identify nodes with their corresponding colors. First consider the case $\deg^+(v) = 0$, implying $v_N = v_E = v_S = v_W = 0$. If any node from $\{v_{NW}, v_{NE}, v_{SE}, v_{SW}\}$ is 0-colored, say v_{NW}, flipping v and not flipping v_N, v_W, and v_{NW} creates a cycle. As v_{NW} is satisfied, the probability for such an event is at least $p(1-p)^2$. Now, assume $v_{NW} = v_{NE} = v_{SE} = v_{SW} = 1$. Then flipping v_N and v_E creates a cycle of 1-nodes. The probability for this to happen is at least p^2.

Let $\deg^+(v) = 1$ and w. l. o. g. assume that v_N is 1-colored. If v_{SW} or v_{SE} is 0-colored, a 0-cycle is created with probability at least $p(1-p)^2$. Hence, assume $v_{SW} = v_{SE} = 1$. If v_{NW} or v_{NE} is 1-colored, say v_{NW}, then v_W is unsatisfied and flipping it and not flipping v_{NW} creates a 1-cycle, with probability $p(1-p)$. The only remaining case is $v_{NW} = v_{NE} = 0$. If the next round flips v and doesn't flip v_W, v_{NW}, v_E, and v_{NE}, then v_N becomes unsatisfied in the following round. Flipping v_N and not flipping v_{NW} creates a cycle. The probability for these two rounds to be successful is at least $p^2(1-p)^5$. □

The expected time to decrease the number of unstable nodes is bounded above by $1/(p^2(1-p)^5) = O(1/p^2)$ if, say, $p \leq 1/2$, hence the following theorem is immediate.

Theorem 7. *The worst-case expected stabilization time for both the max-model and the min-model on $G_{r\times s}$ is $O(n/p^2)$ if $p \leq 1/2$.*

We believe that with random initialization the expected stabilization time is much smaller. It is very unlikely that random initialization creates long paths of unstable 1-nodes. However, such paths of length $\Theta(\log n)$ are still quite likely. Using the same arguments leading to Theorem 6, a lower bound of $\Omega((\log n)/p)$ can be shown. An upper bound is more difficult. We present a bound that is of order $O((\log n)/p^2)$ if the number of rows (or, symmetrically, the number of columns) is constant (and $p \leq 1/2$).

Theorem 8. *After random initialization, the expected stabilization time for both the max-model and the min-model on $G_{r\times s}$ is $O((\log n) \cdot 2^r/p^2)$ if $p \leq 1/2$.*

Proof. Let $L_i := \{(x,i) \mid 0 \leq x \leq r-1\}$, $1 \leq i \leq s$, be the nodes in the i-th column in the graph and note $|L_i| = r$. The probability that all nodes in L_i are initialized zero (or initialized one) is exactly 2^{-r+1}. In this case, L_i is a stable set. The probability that there is no stable set among the consecutive columns $L_i, L_{i+1}, \ldots, L_{i+\gamma-1}$, where $\gamma = 2 \cdot 2^{r-1} \cdot \ln n$ for a fixed i is

$$\left(1 - 2^{-r+1}\right)^{\gamma} = \left(1 - 2^{-r+1}\right)^{2 \cdot 2^{r-1} \cdot \ln n} \leq n^{-2}.$$

Dividing the torus into blocks containing γ consecutive columns each, the probability that each block contains at least one stable column is at least $1 - n^{-1}$. Assume that every block contains a stable column and denote by S the set of stable nodes after initialization. Then $G \backslash S$ consists of connected components, each of which consists of at most $2r\gamma$ nodes. Consider one component C. If two subsequent rounds turn an unsatisfied node in C into a stable node, we speak of a success. Unless C is stable, there is at least one unsatisfied node in C and by Lemma 2 the success probability in two rounds is at least $q := p^2(1-p)^5$. We now argue that with high probability C becomes stable within $2T$ rounds, $T := 4r\gamma/q$. Imagine a sequence of coin flips where each coin shows heads with probability q. By the Chernoff bound the probability that less than $2r\gamma$ out of T coins show heads is at most

$$\exp(-qT/8) = \exp(-r\gamma/2) \leq n^{-2}$$

as $r \geq 2$. As $|C| \leq 2r\gamma$, the probability that C does not become stable within $2T$ rounds is at most n^{-2}. Taking the union bound over at most n components, the whole graph is stable after $2T$ rounds with probability at least $1 - n^{-1}$.

The unconditional probability that the bound $2T$ holds is at least $1 - 2n^{-1}$. In case there is a block without stable column or in case the system has not stabilized after $2T$ rounds, we use the upper bound $O(n/p^2)$ by Theorem 7 to estimate the remaining stabilization time. As this is only necessary with probability at most $2n^{-1}$, the unconditional expected stabilization time is bounded by $2T + O(1/p^2) = O((\log n) \cdot 2^r/p^2)$. □

The bound from Theorem 8 depends crucially on r. However, we do not believe that the stabilization time is significantly affected by the aspect ratio of the torus. Instead, we conjecture that an upper bound $O((\log n)^k/p^k)$ for some $k = O(1)$ holds for all torus graphs.

5.3 Hypercubes

Recall that the node set of a d-dimensional hypercube is given by $\{0,1\}^d$ and edges are between nodes which differ in exactly one coordinate. We are interested in the worst-case expected stabilization time on hypercubes. For torus graphs we identified paths of unstable 1-nodes that delay the stabilization process. As nodes in the d-dimensional hypercube have larger degree if $d > 4$, we identify larger structures of unstable nodes.

Theorem 9. *The worst-case expected stabilization time for both the max-model and the min-model on a d-dimensional hypercube with $n = 2^d$ nodes, $d \geq 4$ even, is $\Omega(n^{1/2} + 1/p)$.*

Proof. As the hypercube is bipartite, it suffices to argue for the min-model. Given a graph $G' = (V', E')$, a *snake-in-box* in G' is a sequence of connected nodes $s'_1, \ldots, s'_\ell$ such that $\{s'_i, s'_j\} \in E'$ implies $j = i \pm 1$ (identifying $s'_{\ell+1}$ with s'_1 and s'_0 with s'_ℓ). It is known how to construct a snake-in-box with length $5/24 \cdot 2^d - 44$ in the d-dimensional hypercube [17]. Let $s_1, \ldots, s_\ell$ be a snake-in-box in the $(d/2)$-dimensional hypercube with $\ell \geq 5/24 \cdot 2^{d/2} - 44$ and let $S = \{s_1, \ldots, s_{\ell-1}\}$. Let $v[i] \in \{0,1\}$ denote the value of the i-th coordinate of v and define an initial cut as follows:

$$v \in V_0(1) \;\Leftrightarrow\; (v[1]v[2]\ldots v[d/2] \in S) \wedge (v[d-1]v[d] = 00).$$

Each 0-node with $v[d-1]v[d] = 00$ is satisfied since flipping one of the last $d/2$ bits results in a 0-neighbor. All other 0-nodes are satisfied since flipping one of the first $d-2 \geq d/2$ bits leads to a 0-neighbor. We conclude that all 0-nodes are satisfied and, therefore, stable. Dividing all 1-nodes into *layers*, layer i contains all 1-nodes v with $v[1]\ldots v[d/2] = s_i$. For a 1-node v flipping a bit at position $i \in \{d/2+1, \ldots, d-2\}$ results in a 1-neighbor. Due to the snake-in-box property of S, v has at most two additional 1-neighbors obtained by flipping single bits among the first $d/2$ positions. More precise, after initialization all 1-nodes in

layers 1 and $\ell - 1$ are unsatisfied with a 1-degree (i. e. number of 1-neighbors) of $d/2 - 1$ while every other 1-node has 1-degree $d/2$ and thus is satisfied. If such an unsatisfied node flips, all its 1-neighbors with 1-degree $d/2$ become unsatisfied.

A layer is called *satisfied* w. r. t. the current cut if it only contains satisfied 1-nodes. Observe that in every round all satisfied layers are connected in the subgraph of all 1-nodes. We focus on the outermost satisfied layers and define as potential the minimum difference $\alpha - \beta$ for $\alpha \leq \beta$ such that for every satisfied layer i we have $\alpha < i < \beta$. Layers α and β therefore "surround" all satisfied layers. The initial potential equals $\ell - 2$ and a potential of 0 is necessary for a stable cut. Layers α and β both contain unsatisfied 1-nodes and a round flipping one of these nodes decreases β or $-\alpha$ by 1, respectively. The probability of decreasing the potential by 1 or 2 in one round is at most $\delta := \min\{1,\ 2^{d/2-1} \cdot p\}$, taking the union bound over at most $2^{d/2-1}$ unsatisfied 1-nodes in layers α and β. The expected waiting time for such an event is bounded below by $1/\delta$, hence the expected time until the potential has decreased to 0 is bounded below by $1/\delta \cdot (\ell - 2)/2 = \Omega(n^{1/2} + 1/p)$. □

6 Conclusions and Future Work

We investigated a self-stabilizing algorithm for maximal and minimal cuts in a restricted distributed environment. The time until the system stabilizes depends on the model of satisfaction, the underlying network, the parameter p, and the initial cut. Surprisingly, the expected stabilization time can range from logarithmic to exponential values. While sparse graphs such as planar graphs, rings, and torus graphs stabilize in expected time $O(n/p^{O(1)})$ (or even in logarithmic time) for max- and min-models, on many dense graphs the stabilization time for the max-model is exponential with high probability if p is constant. Moreover, we have seen for certain torus graphs that there is an exponential gap between random and worst-case initialization.

Several open questions remain, for example a tight bound on the expected stabilization time for all torus graphs and hypercubes with random initialization. Our models use a fixed probability p for flipping unsatisfied nodes. One may also think of other, local strategies, for example flipping an unsatisfied node v with probability proportional to $1/\deg(v)$ or depending on the degrees of v's neighbors.

Acknowledgment. We would like to thank Martin Gairing for helpful comments on an earlier version of this paper.

References

1. Chen, B., Matsumoto, M., Wang, J., Zhang, Z., Zhang, J.: A short proof of Nash-Williams' theorem for the arboricity of a graph. Graphs and Combinatorics 10(1), 27–28 (1994)
2. Dasgupta, A., Ghosh, S., Tixeuil, S.: Selfish stabilization. In: Stabilization, Safety, and Security of Distributed Systems (2006)

3. Diestel, R.: Graph Theory. Springer, Heidelberg (2005)
4. Dijkstra, E.W.: Self-stabilizing systems in spite of distributed control. Communications of the ACM 17(11), 643–644 (1974)
5. Elkin, M.: Distributed approximation: a survey. SIGACT News 35(4), 40–57 (2004)
6. Gairing, M., Goddard, W., Hedetniemi, S.T., Kristiansen, P., McRae, A.A.: Distance-two information in self-stabilizing algorithms. Parallel Processing Letters 14(3-4), 387–398 (2004)
7. Ghosh, S., Karaata, M.H.: A self-stabilizing algorithm for coloring planar graphs. Distributed Computing 7(1), 55–59 (1993)
8. Goddard, W., Hedetniemi, S.T., Jacobs, D.P., Srimani, P.K.: Self-stabilizing protocols for maximal matching and maximal independent sets for ad hoc networks. In: 17th International Parallel and Distributed Processing Symposium (IPDPS 2003), p. 162. IEEE Computer Society, Los Alamitos (2003)
9. Gradinariu, M., Tixeuil, S.: Self-stabilizing vertex coloration and arbitrary graphs. In: Procedings of the 4th International Conference on Principles of Distributed Systems, OPODIS 2000, pp. 55–70 (2000)
10. He, J., Yao, X.: A study of drift analysis for estimating computation time of evolutionary algorithms. Natural Computing 3(1), 21–35 (2004)
11. Hedetniemi, S.T., Jacobs, D.P., Srimani, P.K.: Maximal matching stabilizes in time $O(m)$. Information Processing Letters 80(5), 221–223 (2001)
12. Hedetniemi, S.T., Jacobs, D.P., Srimani, P.K.: Linear time self-stabilizing colorings. Information Processing Letters 87(5), 251–255 (2003)
13. Huang, S.-T., Hung, S.-S., Tzeng, C.-H.: Self-stabilizing coloration in anonymous planar networks. Information Processing Letters 95(1), 307–312 (2005)
14. Kosowski, A., Kuszner, Ł.: Self-stabilizing algorithms for graph coloring with improved performance guarantees. In: Rutkowski, L., Tadeusiewicz, R., Zadeh, L.A., Żurada, J.M. (eds.) ICAISC 2006. LNCS (LNAI), vol. 4029, pp. 1150–1159. Springer, Heidelberg (2006)
15. Manne, F., Mjelde, M., Pilard, L., Tixeuil, S.: A new self-stabilizing maximal matching algorithm. In: Prencipe, G., Zaks, S. (eds.) SIROCCO 2007. LNCS, vol. 4474, pp. 96–108. Springer, Heidelberg (2007)
16. Sauerwald, T., Sudholt, D.: Self-stabilizing cuts in synchronous networks. Technical Report CI-244/08, Collaborative Research Center 531, Technische Universität Dortmund (2008)
17. Tovey, C.A.: Local improvement on discrete structures. In: Local search in combinatorial optimization, pp. 57–89. Princeton University Press, Princeton (1997)