1-極大独立集合問題を解く反復合成を用いた自己安定アルゴリズム

田中 秀幸 * 首藤 裕一 * 角川 裕次 † 増澤 利光 *

内容梗概

ネットワークの 1-極大独立集合を求める自己安定分散アルゴリズムを提案する．1-極大独立集合とは，無向グラフ (つまり，ネットワーク)$G = (V, E)$ の極大独立点集合であって，任意の頂点 $u \in S, v, w \notin S$ $(v \neq w)$ について $S \cup \{v, w\} \setminus \{u\}$ が独立点集合とならないような頂点集合 S を求める問題である．この問題に対して，グラフの各頂点が一意の識別子を持つという仮定のもと動作するサイレント (silent) な自己安定分散アルゴリズムを提案する．提案アルゴリズムは弱公平分散デーモンのもとで動作する．n をグラフの頂点数，D をグラフの直径とすると，収束時間は $O(nD)$ であり，1 頂点あたりの空間計算量は $O(\log n)$ ビットである．これらのアルゴリズムの設計にあたって，反復合成と呼ばれる手法を用いる．

1　はじめに

分散システムとは，多数の計算機 (**ノード**) とそれらを繋ぐ通信リンクから構成されるシステムである．各ノードは互いに通信を行い，自律的かつ協調的に動作することにより問題を解決する．分散システムにおける問題を解くアルゴリズムを**分散アルゴリズム**と呼ぶ．

近年分散システムの大規模化が進むにつれて局所的な故障が無視できない頻度で発生することが多く，多くの分散システムで避けられなくなっている．そのため，分散システムに耐故障性を与えることが重要であると考えられている．また，モバイルノードを含む動的ネットワークでは，ネットワーク形状の動的変化に対する適応性を実現することが重要である．これらの耐故障性や適応性を実現する手法の 1 つとして，自己安定（分散）アルゴリズムとよばれる効果的な解決策が存在し，多くの研究がなされている．

自己安定アルゴリズムとは，任意の初期状況からアルゴリズムの実行を開始しても，やがて問題の要求を満たす状況に収束するという性質を持つアルゴリズムである．自己安定アルゴリズムは，故障やネットワーク形状の変化が発生することによって分散システムがいかなる状況に遷移しても，やがて正常な状況に回復し，次の故障や形状変化が発生するまでの間，正常な状況を保持することができる．したがって，自己安定アルゴリズムは極めて高い耐故障性を持つといえる．

本稿では，極大独立集合問題を拡張した 1-極大独立集合問題を考える．グラフ (ネットワーク)$G = (V, E)$ において，S に属するどの 2 頂点 (2 ノード) も隣接しないような集合 $S \subseteq V$ を独立集合と言う．分散システムの様々なアプリケーションにおいて，より要素数の大きい独立集合を求めることは重要である．独立集合の利用例として，無線ネットワークにおけるクラスタリングなどが挙げられる [1]．しかしながら，**最大独立集合**を求めることは NP 困難である [11]．したがって，**極大独立集合** (MIS) を求める研究が多くなされている．極大独立集合とは，どの頂点 $v \in V \setminus S$ についても $S \cup \{v\}$ が独立集合とならないような集合 S のことである．残念なことに，極大独立集合は常に大きな要素数を持つとは言えない．例えば，スターグラフ (直径 2 の木) において，極大独立集合の要素数は 1 になりうる．したがって，Bollobás et al. [2] により導入された，より強い極大性を持つ 1-極大独立集合 (1-MIS) を考える．極大独立集合 $S \subseteq V$ は，任意の頂点 $u \in S, v, w \notin S$ $(v \neq w)$ について $S \cup \{v, w\} \setminus \{u\}$ が独立集合とならないならば，1-極大独立集合である．スターグラフの場合，n 個の頂点からなるスターの 1-極大独立集合の要素数は $n - 1$ となる．

1.1　関連研究

極大独立集合問題はグラフ理論や分散システムにおいて基本的な問題の 1 つであるため，数多くの研究がなされてきた．表 1 に，MIS や 1-MIS に対する自己安定アルゴリズムの最近の結果を示す．1995 年に，Shukla et al.[10] により，任意のグラフに対する自己安定 MIS アルゴリズムが提案された．そのアルゴリズムは，集中デーモンを仮定しており，収束時間は $O(n)$ ステップ，1 頂点あたり 1 ビット (2 状態) で最適である．ここで，集中デーモンとは，各ステップにおいてただひとつの頂点にのみアクションを実行させるスケジューラである．このアルゴリズム匿名ネットワーク上で動作する．すなわち，頂点が一意な識別子を持つ必要がない．Ikeda et al.[7] では，MIS に対する別の自己安定アルゴリズムが提案された．そのアルゴリズムは，各頂点に識別子が存在することを仮定するが，分散デー

*大阪大学大学院 情報科学研究科
†龍谷大学 理工学部 数理情報学科

表 1: MIS と 1-MIS に対する自己安定アルゴリズム．n と D はそれぞれグラフの全頂点数と直径を表す．

	問題	グラフ	ID	スケジューラ	収束時間		空間計算量
					ステップ	ラウンド	
[10]	MIS	任意	なし	集中デーモン	$O(n)$	$O(n)$	$O(1)$ bits
[7]	MIS	任意	あり	分散デーモン	$O(n^2)$	$O(n)$	$O(1)$ bits
[11]	MIS	任意	あり	分散デーモン	$O(n)$	$O(n)$	$O(1)$ bits
[9]	1-MIS	木	なし	集中デーモン	$O(n^2)$	$O(n)$	$O(1)$ bits
[8]	1-MIS	任意	あり	集中デーモン	-	$O(n^2)$	$O(n \log n)$ bits
Proposed	1-MIS	任意	あり	分散デーモン	-	$O(nD)$	$O(\log n)$ bits

モンのもとで正しく動作する．ここで，分散デーモンとは，各ステップにおいて複数の頂点がアクションを実行できるようなスケジューラである．1 頂点あたりの空間計算量は 1 ビット (2 状態) で最適であるが，収束時間は $O(n^2)$ ステップである．Turau[11] は空間計算量を $O(1)$ ビット (3 状態) に増やすことで，収束時間を $O(n)$ ステップに改善している．

Shi et al.[9] は，初めて 1-MIS に対する自己安定アルゴリズムを提案した．そのアルゴリズムはグラフが木であることを仮定しており，集中デーモンのもとで正しく動作する．収束時間は $O(n^2)$ ステップであり，1 頂点あたりの空間計算量は $O(1)$ ビットである．難波 [8] は，任意のグラフにおいて，集中デーモンのもとで正しく動作する，1-MIS に対する自己安定アルゴリズムを提案した．論文中で，収束時間に関する解析は示されていないが，収束時間は $O(n^2)$(非同期) ラウンドである．難波のアルゴリズムは，部分アルゴリズムを並列に n 個実行するため，1 頂点あたりの空間計算量は $O(n \log n)$ ビットである．

各頂点に識別子が存在しない場合 (匿名グラフ)，初期状況における対称性を壊すことができないため，任意のグラフにおいて，分散デーモンのもとで動作する，MIS(1-MIS) に対する決定性アルゴリズムは存在しない．

1.2 本研究の成果

本稿では，極大独立集合よりも極大性の高い，1-極大独立集合問題を考える．この問題に対して，各頂点が一意な識別子を持つ，任意な連結無向グラフで動作するサイレントな自己安定アルゴリズムを提案する．n をグラフの頂点数，D をグラフの直径とすると，収束時間は $O(nD)$(非同期) ラウンドであり，1 頂点あたりの空間計算量は $O(\log n)$ である．このアルゴリズムの設計にあたって，反復合成と呼ばれる手法を用いる．

2 諸定義

ネットワークは単純連結無向グラフ $G = (V, E)$ で表される．V はネットワーク内の頂点の集合，E はネットワーク内の辺の集合とする．ネットワークの頂点数を $n = |V|$ とする．各頂点は，非負整数の集合 ID から選ばれる一意な識別子 id を持つ．頂点 v に接続する辺と繋がっている頂点を，頂点 v の隣接頂点と呼び，頂点 v の隣接頂点の集合を $N_v = \{w \in V \mid \{w, v\} \in E\}$ と表す．

通信モデルとして局所共有メモリモデル [5] を用いる．頂点は有限状態機械によりモデル化され，その状態は頂点が有する変数の値により決まる．ある頂点は自身とその隣接頂点の変数の値を同時に読み込むことができるが，書き換え可能なのは自身の変数のみである．

各頂点のアルゴリズムをアクション $< label >< guard > \longrightarrow < statement >$ の有限集合により定義する．$label$ は参照のため，およびアクションの優先度を表すために使用する．$guard$ は頂点 v とその隣接頂点の変数と識別子に関する述語である．$statement$ は頂点 v の 1 つ以上の変数の値を更新する処理を定める．頂点 v の 1 つ以上のアクションの $guard$ が真であるとき，v は**実行可能**であるという．$gurad$ の真偽の決定と対応する $statement$ の実行は 1 原子ステップで起こると仮定する [6]．

ネットワークの**状況**は各頂点の状態から成るベクトルで表される．状況 γ における，頂点 v の変数 x の値を $\gamma(v).x$ で表す．**ステップ**と呼ばれる，各状況の変遷は**デーモン**により引き起こされる．本稿では**分散型デーモン**を仮定する．分散型デーモンとは，各ステップで 1 つ以上の任意の実行可能な頂点のアクションを同時に実行できるデーモンである．実行される頂点に 2 つ以上の実行可能なアクションがある場合，最も優先度が高いラベルのアクションを実行する．アルゴリズム $\mathcal{A}$ の 1 つのステップにより状況が γ から γ' に変わることを $\gamma \mapsto_{\mathcal{A}} \gamma'$ と表す．アルゴリズム $\mathcal{A}$ の**実行**とは，全ての $i \geq 0$ に対して $\gamma_i \mapsto_{\mathcal{A}} \gamma_{i+1}$ となるような状況の系列 $\gamma_0, \gamma_1, \cdots$ である．デーモンは**弱公平**であることを仮定する．すなわち，連続して実行可能な頂点はいずれ実行されることが保証される．

自己安定アルゴリズムとは，ネットワークのどのような状況から実行を開始しても，いずれ正しい動作に復帰することを保証するアルゴリズムである．つまり，各頂点はどのような状態からアルゴリズムの実行を開始してもよい．状況 γ において実行可能な頂点が存在しないとき，γ を**最終状況**と呼ぶ．無限長の実行および有限長で末尾の状況が最終状況であるような実行を**極大実行**と呼ぶ．$\mathcal{L}$ を状況に対する述語とする．アルゴリズム $\mathcal{A}$ の極大実行 $\gamma_0, \gamma_1, \cdots$ が次の (i),(ii) を満たすとき，$\mathcal{A}$ は $\mathcal{L}$ に対して自己安定である．(i) ある $i \geq 0$ に対して，$\mathcal{L}(\gamma_i)$，(ii) 全ての $j \geq i$ に対して，$\mathcal{L}(\gamma_j)$．(i) は**収束性**，(ii) は**閉包性**と呼ばれる．$\mathcal{A}$ の任意の極大実行が有限であるとき，$\mathcal{A}$ は**サイレント**であるという．

実行の収束時間は非同期ラウンドで測定する．頂点 v が状況 γ_i で実行可能で，状況 γ_{i+1} で実行可能でないとき，頂点 v はステップ $\gamma_i \mapsto \gamma_{i+1}$ で**無効化**されたという．実行 $\varrho = \gamma_0, \gamma_1, \ldots$ の最初のラウンドを，γ_0 で実行可能な全頂点が一度は実行される，もしくは無効化される最小の接頭辞とし，$\gamma_0 \ldots \gamma_s$ と表す．ϱ の 2 番目のラウンドを，実行 $\gamma_s, \gamma_{s+1}, \ldots$ の最初のラウンドとする．それ以降のラウンドも同様に定義する．ϱ の実行時間をそのラウンドの数として測定する．

2.1 問題定義

本稿では 1-極大独立集合問題を考える．連結無向グラフ $G = (V, E)$ において，頂点の部分集合 $S \subset V$ の互いに異なる 2 つの頂点が隣接しないとき，S を**独立集合**と呼ぶ．また，どの頂点 $v \in V \setminus S$ についても $S \cup \{v\}$ が独立集合とならないような独立集合 S を**極大独立集合**と呼ぶ．さらに，極大独立集合 S に対して，S に含まれるどの頂点を 1 つ取り除き，S には含まれないどの 2 頂点を追加しても独立集合にはならないとき，集合 S を *1*-**極大独立集合**と呼ぶ．つまり，1-極大独立集合 S とは，任意の頂点 $u \in S, v, w \notin S$ $(v \neq w)$ について $S \cup \{v, w\} \setminus \{u\}$ が独立集合とならないような極大独立集合である．*1*-**極大独立集合問題**では，各頂点 v が変数 $v.\texttt{mis} \in \{\textbf{true}, \textbf{false}\}$ を持ち，頂点集合 $\{v \in V \mid v.\texttt{mis}\}$ が 1-極大独立集合となるように各頂点 v の $v.\texttt{mis}$ を設定する問題である．ここで変数 `mis` が真の頂点を独立頂点と呼ぶこととする．$\mathcal{L}_{1\mathrm{MIS}}$ を，状況 γ において，$\{v \in V \mid v.\texttt{mis} = \textbf{true}\}$ が 1-極大独立集合であるとき，かつそのときのみ $\mathcal{L}_{1\mathrm{MIS}}(\gamma) = \textbf{true}$ となる，状況に対する述語とする．本稿では $\mathcal{L}_{1\mathrm{MIS}}$ に対してサイレントな自己安定であるアルゴリズムを与える．

3 反復合成

反復合成 [3] とは，アルゴリズム $\mathcal{A}$，$\mathcal{P}$ と述語 E を入力として述語 $\mathcal{L}$ に対するサイレントな自己安定アルゴリズム $\mathbf{Loop}(\mathcal{A}, E, \mathcal{P})$ を生成するフレームワークである．反復合成を利用するためには，与えられた述語 $\mathcal{L}$ に対して，後述する条件を満たすように $\mathcal{A}$，$\mathcal{P}$，述語 E を設計しなければならない．アルゴリズム $\mathcal{A}$ は，繰り返し実行される基盤アルゴリズムである．$\mathcal{A}$ は後述する 3 つの条件 (*shiftable convergence*，*loop convergence*，*correctness*) を満たす必要がある．述語 $E : V \mapsto \{\textbf{false}, \textbf{true}\}$ は，ローカルにエラー検出を行う述語である．ある頂点 $v \in V$ で $E(v)$ が成り立つような状況 γ を，**エラー状況**と呼ぶ．全ての頂点で $v \in V$ で $\neg E(v)$ が成り立つような状況 γ を，**非エラー状況**と呼ぶ．アルゴリズム $\mathcal{P}$ は，任意の状況から，ネットワークを非エラー状況にするサイレントな自己安定アルゴリズムである．大まかに言うと，$\mathbf{Loop}(\mathcal{A}, E, \mathcal{P})$ は次のような動作をする．ここで，先述の通り，状況 γ において実行可能な頂点が存在しないとき，かつそのときのみ γ は $\mathcal{A}$ の最終状況である．

1: **repeat**
2: **if** 現在の状況がエラー状況である **then**
3: $\mathcal{P}$ を実行 (これにより，ネットワークが非エラー状況になる)
4: **else**
5: $\mathcal{A}$ を実行 (これにより，ネットワークは $\mathcal{A}$ の最終状況になる)
6: $\mathcal{A}$ の出力を $\mathcal{A}$ の入力にコピー
7: **end if**
8: **until** 現在の状況が $\mathcal{A}$ の最終状況かつ，$\mathcal{A}$ の入力と出力が同じである

以下では，$\mathcal{A}$，$\mathcal{P}$ が満たすべき条件と，$\mathcal{A}$ の出力を $\mathcal{A}$ の入力へ**コピー**することの意味について説明する．$\mathcal{A}(\mathrm{resp.}\mathcal{P})$ のアクションにより値が更新される変数の集合を $O_{\mathcal{A}}(\mathrm{resp.}O_{\mathcal{P}})$ とする．$\mathcal{A}(\mathrm{resp.}\mathcal{P})$ のアクションにより値が更新されることはなく，値が参照されるだけの変数の集合を $I_{\mathcal{A}}(\mathrm{resp.}I_{\mathcal{P}})$ とする．$O_{\mathcal{A}} \cap O_{\mathcal{P}} = \emptyset$，$I_{\mathcal{P}} = \emptyset$ と仮定する．エラー検出述語 $E(v)$ は頂点 $v \in V$ により評価され，その真偽が自身と自身の隣接頂点の $I_{\mathcal{A}} \cup O_{\mathcal{P}}$ の変数にのみ基づく述語である．$\mathcal{E}$ を，状況 γ で $\bigvee_{v \in V} E(v)$ が成り立つとき，かつそのときのみ $\mathcal{E}(\gamma)$ が成り立つような，状況に対する述語とする．アルゴリズム $\mathcal{A}$ は全ての変数 $x \in O_{\mathcal{A}}$ に対して，**コピー変数** $\overline{x} \in I_{\mathcal{A}}$ を持つと仮定する．γ^{copy} を，状況 γ で，全頂点 v とその全ての変数 $x \in O_{\mathcal{A}}$ に対して $v.\overline{x}$ の値を $v.x$ の値で置き換えて得られる状況とする．$\gamma \in \neg\mathcal{E}$，$\gamma^{copy} = \gamma$ が成り立ち，どの頂点でも全アクションが実行可能でないとき，状況 γ は述語 $\mathcal{C}_{goal}(\mathcal{A}, E)$ を満たすとする．アルゴリズム $\mathcal{A}$ は，ある整数 $R_{\mathcal{A}}$ と $L_{\mathcal{A}}$ に対して，次の 3 つの条件を満たすように設計されている必要がある．

Shiftable Convergence 状況 $\neg\mathcal{E}$ から開始される $\mathcal{A}$ の任意の極大実行は，$\gamma^{copy} \in \neg\mathcal{E}$ であるような状況 γ で終了する．

Loop Convergence $\varrho_0, \varrho_1 \ldots$ が，各 $i \geq 0$ に対して $\varrho_i = \gamma_{i,0}, \gamma_{i,1}, \ldots, \gamma_{i,s_i}$，$\gamma_{0,0} \in \neg\mathcal{E}$，$\gamma_{i+1,0} = \gamma_{i,s_i}^{copy}$ であるような，アルゴリズム $\mathcal{A}$ の無限の系列ならば，ある $j < \mathcal{L}_{\mathcal{A}}$ に対して，$\gamma_{j,s_j} \in \mathcal{C}_{\mathrm{goal}}(\mathcal{A}, E)$ と $R(\varrho_0) + R(\varrho_1) + \cdots + R(\varrho_j) \leq R_{\mathcal{A}}$ が成り立つ．

Correctness 任意の状況 γ に対して，$\gamma \in \mathcal{C}_{goal}(\mathcal{A}, E) \Rightarrow \gamma \in \mathcal{L}$ が成り立つ．

$L_{\mathcal{A}}$ は $\mathcal{A}$ の実行回数の上界を表し，$R_{\mathcal{A}}$ は $\mathcal{A}$ の繰り返しの実行における全ラウンド数の上界を表す．$\mathcal{P}$ の任意の極大実行は，$\neg\mathcal{E}$ であるような状況で $T_{\mathcal{P}}$ ラウンド以内に終了するように設計されている必要がある．

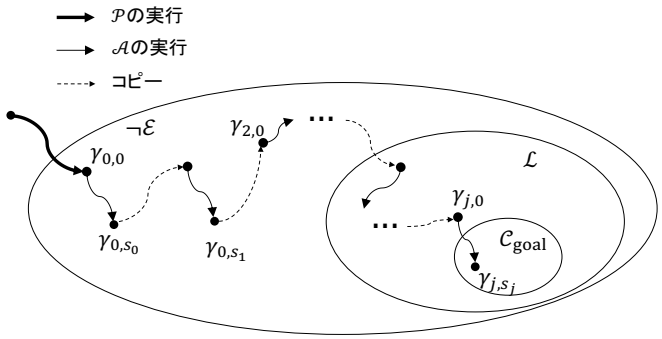

図 1: $\mathbf{Loop}(\mathcal{A}, E, \mathcal{P})$ の実行

上記の条件を満たすように $\mathcal{A}$，$\mathcal{P}$，述語 E を設計したならば，作成されたアルゴリズム $\mathbf{Loop}(\mathcal{A}, E, \mathcal{P})$ において以下の定理が導かれる

定理 3.1 ([4][1])．アルゴリズム $\mathbf{Loop}(\mathcal{A}, E, \mathcal{P})$ は述語 $\mathcal{L}$ に対してサイレントであり，自己安定である．任意の $\mathbf{Loop}(\mathcal{A}, E, \mathcal{P})$ の実行は $O(n + T_{\mathcal{P}} + R_{\mathcal{A}} + L_{\mathcal{A}}D)$ ラウンドで終了する．$\mathbf{Loop}(\mathcal{A}, E, \mathcal{P})$ の 1 頂点あたりの空間計算量は $O(S_{\mathcal{A}} + S_{\mathcal{P}} + \log n)$ ビットである．$S_{\mathcal{A}}$，$S_{\mathcal{P}}$ はそれぞれ，アルゴリズム $\mathcal{A}$，$\mathcal{P}$ の 1 頂点あたりの空間計算量である．

以降で $\mathbf{Loop}(\mathcal{A}, E, \mathcal{P})$ の動作について簡単に説明する．$\mathbf{Loop}(\mathcal{A}, E, \mathcal{P})$ の実行において，各頂点はアルゴリズム $\mathcal{A}$ かアルゴリズム $\mathcal{P}$ のどちらかを実行する．任意の状況から始まったとしても，やがて全頂点は $\mathcal{A}$ と $\mathcal{P}$ のどちらを実行するのかについて合意を達する．現在の状況が $\neg\mathcal{E}$ を満たさないと検知されたとき，全頂点はアルゴリズム $\mathcal{P}$ を実行し，$\neg\mathcal{E}$ を満たす状況にする (図 1 の $\mathcal{P}$ の実行の矢印)．現在の状況が $\neg\mathcal{E}$ を満たすとき，全頂点はアルゴリズム $\mathcal{A}$ を実行する (図 1 の $\mathcal{A}$ の実行の矢印)．$\mathcal{A}$ の実行が終了する度に，状況 $\gamma \in \mathcal{C}_{goal}(\mathcal{A}, E)$ に到達しているかを確認する．もし，到達していたならば，その後何も動作しない．到達していないならば，全ての出力変数の値を対応するコピー変数にコピーする．それにより，現在の状況から別の状況へ変遷する (図 1 のコピーの矢印)．その後，再び新たに $\mathcal{A}$ を実行する．$\mathcal{A}$ を設計する上での条件である Shiftable Convergence と Loop Convergence より，やがて $\mathbf{Loop}(\mathcal{A}, E, \mathcal{P})$ の実行は状況 $\gamma \in \mathcal{C}_{goal}(\mathcal{A}, E)$ に到達し，終了する．その後，$\mathcal{A}$ を設計する上での条件である Correctness により述語 $\mathcal{L}$ は常に成り立つ．

4 自己安定 1-極大独立集合アルゴリズム

本節では，前節で示した反復合成を用いて，述語 $\mathcal{L}_{1MIS}$ に対するサイレントな自己安定であるアルゴリズムを提案する．具体的には，反復合成アルゴリズム $\mathbf{Loop}(Inc, E_{\text{MIS}}, Init)$ が述語 $\mathcal{L}_{1MIS}$ に対するサイレントな自己安定アルゴリズムとなるような基盤アルゴリズム Inc，初期化アルゴリズム $Init$，エラー検出述語 E_{MIS} を本節で与える．$\text{Loop}(Inc, E_{MIS}, Init)$ において，1 頂点あたりの空間計算量は $O(\log n)$ ビットであり，収束時間は $O(nD)$ ラウンドである．

基盤アルゴリズム Inc において，各頂点 v は変数 $v.\texttt{mis} \in \{\mathbf{false}, \mathbf{true}\}$ とそれに対応するコピー変数 $v.\overline{\texttt{mis}} \in \{\mathbf{false}.\mathbf{true}\}$ を持つ．$v.\overline{\texttt{mis}}$ は $Init$ の出力変数であり，Inc の入力変数である．$v.\texttt{mis}$ は $Init$ では使用されず，Inc の出力変数である．ここで，$S_I = \{v \in V \mid v.\overline{\texttt{mis}}\}$，$S_O = \{v \in V \mid v.\texttt{mis}\}$ とする．S_I は Inc の入力における全ての独立頂点からなる集合であり，S_O は Inc の出力における全ての独立頂点からなる集合である．$\mathcal{L}_{input}$ を，状況 γ において S_I が MIS である，かつそのときのみ $\mathcal{L}_{input}(\gamma) = \mathbf{true}$ が成り立つような，状況に対する述語とする．ここで，$\mathcal{L}_{input}$ は前節における $\neg\mathcal{E}$ に対応する．

本節では以下を満たすような Inc，E_{MIS}，$Init$ を与える．

- S_I が MIS でないならば，少なくとも 1 つの頂点 v において $E_{MIS}(v) = \mathbf{true}$ が成り立つ．すなわち，状況 γ において $\neg\bigvee_{v\in V} E_{MIS}(v)$ が成り立つとき，かつそのときのみ $\mathcal{L}_{input}(\gamma) = \mathbf{true}$ が成り立つ．
- 任意の状況から開始する $Init$ の全ての極大実行は，$\mathcal{L}_{input} = \mathbf{true}$ が成り立つような状況で $O(n)$ ラウンド以内に終了する．
- $\mathcal{L}_{input} = \mathbf{true}$ であり，S_I が 1-MIS でないような状況から開始する Inc の極大実行 ϱ は，S_O が MIS であり $|S_O| \geq |S_I| + 1$ が成り立つような状況で $O(\epsilon + 1)$ ラウンド以内に終了する．ここで，ϵ は実行 ϱ の最終状況における $|S_O| - |S_I|$ である．
- $\mathcal{L}_{input} = \mathbf{true}$ であり，S_I が 1-MIS であるような状況から開始する Inc の極大実行 ϱ は，S_I が 1-MIS であり，$S_O = S_I$ であるような状況で $O(1)$ ラウンド以内に終了する．

[1]反復合成のフレームワーク $\mathbf{Loop}(\mathcal{A}, E, \mathcal{P})$ 自体は [3] で提案されたが，[4] でその計算量の解析が改善され，定理 3.1 が証明された．

上記が全て成り立てば，Inc，E_{MIS}，$Init$ は，前節の反復合成を使用する上での条件を全て満たし，$\mathcal{L} = \mathcal{L}_{1MIS}$，$T_{Init} = O(n)$，$R_{Inc} = O(n)$，$L_{Inc} = n$ となる．したがって，$\mathbf{Loop}(Inc, E_{\mathrm{MIS}}, Init)$ は述語 $\mathcal{L}_{1MIS}$ に対してサイレントな自己安定アルゴリズムであり，収束時間は $O(n + T_{Init} + R_{inc} + L_{inc} \cdot D) = O(nD)$ ラウンドである．

4.1 エラー検出述語 E_{MIS} と初期化アルゴリズム $Init$

エラー検出述語を次の通り与える．

$$E_{\mathrm{MIS}}(v) \equiv (v.\overline{\texttt{mis}} \wedge \exists u \in N_v : u.\overline{\texttt{mis}}) \vee (\neg v.\overline{\texttt{mis}} \wedge \forall w \in N_v : \neg w.\overline{\texttt{mis}}).$$

補題 4.1. 任意の状況 γ において，$\neg \bigvee_{v \in V} E_{\mathrm{MIS}}(v)$ が成り立つとき，かつそのときのみ $\mathcal{L}_{\mathrm{input}}(\gamma)$ が成り立つ．

Proof. まず $\neg E_{\mathrm{MIS}}(v) \equiv (v.\overline{\texttt{mis}} \Rightarrow \forall u \in N_v : \neg u.\overline{\texttt{mis}}) \wedge (\neg v.\overline{\texttt{mis}} \Rightarrow \exists w \in N_v : w.\overline{\texttt{mis}})$ である．ある状況 γ で $\neg \bigvee_{v \in V} E_{\mathrm{MIS}}(v)$ が成り立つとする．それは，状況 γ において全ての頂点 $v \in V$ で $\neg E_{\mathrm{MIS}}(v)$ が成り立つということを意味する．そのとき，S_I に属す全ての頂点は，S_I に属すような隣接頂点を持たず，S_I に属さない全ての頂点は S_I に属すような隣接頂点を少なくとも 1 つ持つ．したがって，状況 γ において S_I は MIS であり，$\mathcal{L}_{\mathrm{input}}(\gamma)$ が成り立つ．ある状況 γ で $\bigvee_{v \in V} E_{\mathrm{MIS}}(v)$ が成り立つとする．その場合，S_I に属すある頂点が，S_I に属す隣接頂点を持つ，もしくは，S_I に属さないある頂点が S_I に属す隣接頂点を持たない．したがって，状況 γ において S_I は MIS でなく，$\mathcal{L}_{\mathrm{input}}(\gamma)$ は成り立たない． □

初期化アルゴリズム $Init$ のアクションを表 1 に示す．$Init$ の実行により，$O(n)$ ラウンド以内に，S_I が MIS であるような状況になる．$Init$ では，小さい識別子を持つ頂点が優先して S_I に属すことになる．ある頂点 v は，S_I に属し v より識別子が小さいような隣接頂点を持たないならば，S_I に追加される．つまり $v.\overline{\texttt{mis}} \leftarrow \mathbf{true}$ を実行する．そうでないならば，つまり v が S_I に属し v より識別子が小さいような隣接頂点を持つならば，$v.\overline{\texttt{mis}} \leftarrow \mathbf{false}$ を実行する．

表 1: $Init$

[**Actions of process** v]			
I_1:	$v.\overline{\texttt{mis}}$	$\longleftarrow$	$(\forall w \in N_v : \neg w.\overline{\texttt{mis}} \vee (v.id < w.id))$

補題 4.2. 任意の状況から始まる $Init$ の全ての極大実行は，$\mathcal{L}_{\mathrm{input}}$ が成り立つような状況で $O(n)$ ラウンド以内で終了する．

Proof. まず，$Init$ の任意の最終状況で $\mathcal{L}_{\mathrm{input}}$ が成り立つことを示す．3 節の"$v.x \longleftarrow \chi(v)$"の定義より，$v.\overline{\texttt{mis}} \neq (\forall w \in N_v : \neg w.\overline{\texttt{mis}} \vee (v.id < w.id))$ であるとき，かつそのときのみ，頂点 v は実行可能である．したがって，実行可能が頂点が存在しないような最終状況 γ において，S_I に属す全ての頂点 v は，S_I に属すような隣接頂点を持たず，S_I に属さない全ての頂点 v は，S_I に属すような隣接頂点を少なくとも 1 つ持つ．以上より，任意の最終状況において，$\mathcal{L}_{\mathrm{input}}$ は成り立つ．

次に $Init$ の全ての極大実行は，$O(n)$ ラウンド以内に終了することを示す．$v_1, v_2, \ldots, v_n$ を，$v_1.id < v_2.id < \cdots < v_n.id$ であるような頂点の集合とする．アクション I_1 のガードは $v.\overline{mis}$ の真偽と，$w.id < v.id$ であるような $w.\overline{mis}$ の真偽でのみ決定される．したがって，$Init$ の任意の極大実行の第 1 ラウンドで $v_1.\overline{\texttt{mis}}$ は決定され，以後，実行可能となることはない．同様に，任意の $i \geq 2$ について，$v_i.\overline{\texttt{mis}}$ は，N_{v_i} に属す v_i より識別子が小さいような全ての頂点の $v_i.\overline{\texttt{mis}}$ が決定されたのち，1 ラウンド以内に決定され，以後実行可能となることはない．以上より，$Init$ の全ての極大実行は，$O(n)$ ラウンド以内に終了する． □

4.2 アルゴリズム Inc

基盤アルゴリズム Inc のアクションを表 2 に示し，表 2 で使用されている関数を表 3 に示す．Inc は S_I が MIS であることを仮定して動作する．このアルゴリズムの実行では，S_I が 1-MIS でないならば，$|S_O| \geq |S_I| + 1$ となるような極大独立集合 S_O を求め，S_I が 1-MIS ならば，$|S_O| = |S_I|$ であるような状況に到達する．図 2 は，アルゴリズム Inc の入力 S_I の 1 例であり，$S_I = \{5, 23, 71\}$ である．ここで，S_I は極大独立集合であるが，1-MIS ではない．

4.2.1 アルゴリズムの概要

この節では，S_I が 1-MIS でないならば，$|S_O| \geq |S_I| + 1$ となるような極大独立集合 S_O を求め，S_I が 1-MIS ならば，$|S_O| = |S_I|$ であるような状況に到達するようなアルゴリズムを与える．Inc のアクションを表 2 に示し，表 2 で使用される関数を表 3 に示す．

まず，頂点の親子関係を定義する．頂点 $v \in V \setminus S_I$ が 1 つの隣接頂点 $u \in S_I$ を持つとき，u は v の親であり，v は u の子であるとする．ここで，頂点 $v \in V \setminus S_I$ が 1 つの隣接頂点 $u \in S_I$ を持つとは，$N_v \wedge S_I = \{u\}$ が成り立つ

Table 2: Inc

[**Actions of process** v]			
M_1:	$v.\texttt{parent}$	$\longleftarrow$	$Parent(v)$
M_2:	$v.\texttt{numchild}$	$\longleftarrow$	$\lvert C_v \rvert$
M_3:	$v.\texttt{cand}$	$\longleftarrow$	$Cand(v)$
M_4:	$v.\texttt{qualifier}$	$\longleftarrow$	$Qualifier(v)$
M_5:	$v.\texttt{winner}$	$\longleftarrow$	$Winner(v)$
M_6:	$v.\texttt{mis}$	$\longleftarrow$	$InS_A(v) \vee InS_B(v) \vee InS_C(v)$

Table 3: Inc の関数

$$Parent(v) = \begin{cases} \min\{w.id \mid w.\overline{\texttt{mis}}\} & \textbf{if } |\{w \in N_v \mid w.\overline{\texttt{mis}}\}| = 1 \\ \bot & \textbf{otherwise} \end{cases}$$

$$C_v = \{w \in N_v \mid w.\texttt{parent} = v\}$$

$$Cand(v) \equiv v.\texttt{parent} \neq \bot \wedge (v.\texttt{parent}).\texttt{numchild} - |\{w \in N_v \mid v.\texttt{parent} = w.\texttt{parent}\}| \geq 2$$

$$Qualifier(v) \equiv v.\texttt{cand} \wedge (\forall w \in N_v : w.\texttt{cand} \Rightarrow v.\texttt{parent} \leq w.\texttt{parent})$$

$$Winner(v) \equiv v.\texttt{qualifier} \wedge (\forall w \in N_v : v.id < w.id \vee \neg w.\texttt{winner})$$

$$InS_A(v) \equiv v.\overline{\texttt{mis}} \wedge |\{w \in C_v \mid w.\texttt{winner}\}| \leq 1$$

$$InS_B(v) \equiv \neg v.\overline{\texttt{mis}} \wedge v.\texttt{winner} \wedge \neg(v.\texttt{parent}).\texttt{mis}$$

$$InS_C(v) \equiv \neg v.\overline{\texttt{mis}} \wedge \neg v.\texttt{winner} \wedge \begin{pmatrix} \forall w \in N_v \ s.t. \ w.\texttt{mis} : \\ \neg\left(w.\overline{\texttt{mis}} \vee w.\texttt{winner} \vee w.id < v.id\right) \end{pmatrix}$$

ことである．この定義により，S_I が 1-MIS でないならば，高さ 1 の木が，1 つ以上生成される．各頂点 v は，親が存在するならば，その親の識別子を $v.\texttt{parent}$ に保存する．また，親を持たないならば，$v.\texttt{parent}$ に $\bot$ を保存する．C_u を頂点 u の子の集合とする．よって $C_u = \{v \in N_u \mid v.\texttt{parent} = u\}$ となる．u と v を，u が v の親であるような任意の 2 つの頂点とする．$|C_u| - |C_v \setminus N_v| \geq 2$ が成り立つとき，v を**候補**と呼ぶ．言い換えると，v は，その親が存在し，その親が $V \setminus N_v$ に属す v 以外の子を持つならば，候補になる．図 4 を用いて具体的に説明する．頂点 31 の親 (頂点 5) は，頂点 31 の隣接頂点でない別の子 (頂点 61) を持つので，頂点 31 は候補である．一方で，頂点 13 は，頂点 13 の親 (頂点 5) の全ての子 (頂点 31 と頂点 62) に隣接しているので，候補でない．言い換えると，候補とは，その親が独立頂点でなくなることにより，その親の子である 2 つ以上の候補が独立頂点になることで，MIS の要素数を増やすことができるような頂点である．例えば，頂点 5 が独立頂点でなくなることで，その隣接する候補である頂点 31 と頂点 62 は独立頂点になることができるので，S_I より要素数が大きい MIS を得ることができる．しかしながら，候補同士が隣接する可能性があり，その隣接している候補の両方を独立頂点にすると，S_O が MIS でなくなるので，全ての候補を独立頂点にすることはできない場合がある．具体的には，頂点 53 と頂点 79 の両方，もしくは頂点 11 と頂点 81 の両方を独立頂点にすることはできない．

上記で述べた，候補が隣接する問題を解決するために 2 つのフィルタ (第 1 フィルタと第 2 フィルタ) を導入する．第 1 フィルタを通過する候補を**予選通過者**，第 2 フィルタを通過する候補を**勝者**とする．候補 v は，$w.\texttt{parent} < v.\texttt{parent}$ が成り立つような隣接する候補 w が存在しないならば，予選通過者になる．図 4 において，頂点 79 は，その親 (頂点 71) の識別子が 71 であり，その隣接頂点 (頂点 53) の親の識別子が 23 なので，予選通過者でない．図 4 の例で，別の親を持ち，隣接している候補が存在しないような候補は予選通過者となる．第 1 フィルタのみでは，同一の親を持ち，隣接している候補同士の問題を解決できていない．よって，第 2 フィルタで，アルゴリズム $Init$ と同じように，それら候補同士の識別子を比較することにより，勝者を決定する．$q_1, q_2, \ldots, q_s$ を，$q_1.id < q_2.id < \ldots q_s.id$ であるような予選通過者であるとする．そして，集合 W を次のように定義する．$q_1 \in W$ であり，各 $i \geq 2$ に対して，q_i に隣接する $j < i$ であるような q_j が存在しないとき，かつそのときのみ $q_i \in W$ である．図 4 の例では，頂点 11，31，37，53，62 が勝者となる．勝者について次の 2 つの補題が成り立つ．

補題 4.3. S_I が 1-MIS でないならば，子に 2 つ以上の勝者が存在するような頂点が少なくとも 1 つ存在する．

Proof. 子に候補が存在するような頂点のうち，最小の識別子である頂点を u をする．S_I が 1-MIS でないならば，そのような u が必ず存在する．c_1 を，C_u に属す候補のうち，最小の識別子である頂点とする．候補の定義より，C_u に属し，c_1 と隣接しないような候補が 1 つ以上存在する．それらの中で，最小の識別子である頂点を c_2 とする．c_1 と c_2 は，その親である u の識別子の最小性により，第 1 フィルタを通過し，c_1 と c_2 自身の識別子の最小性により，第 2 フィルタを通過する．以上より，u の子に少なくとも 2 つの勝者が存在する． □

補題 4.4. S_I が 1-MIS でないならば，勝者は存在しない．

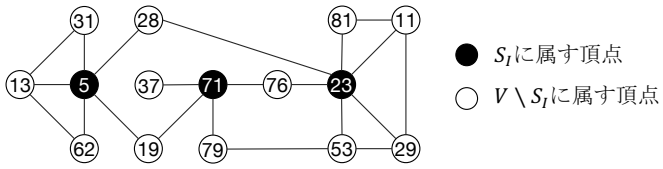

図 2: S_I の 1 例

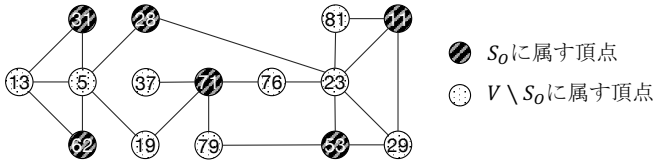

図 3: S_I が図 2 で示されるような Inc の 1 回の実行により導かれる S_O

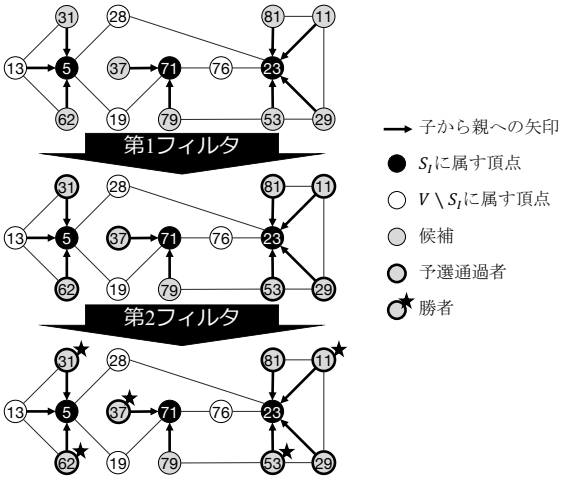

図 4: 第 1 フィルタと第 2 フィルタ

Proof. S_I が 1-MIS であり，勝者 $v \in V$ が存在すると仮定して，背理法で証明する．u を v の親とする．候補の定義より，u の子の中に，v に隣接しないような別の候補が少なくとも 1 つ存在する．そのような候補を w とする．v と w は候補なので，$S_I \cap (N_v \cup N_w) = \{u\}$ が成り立つ．したがって，$S_O = S_I \cup \{v, w\} \setminus \{u\}$ は独立集合であり，$|S_O| = |S_I| + 1$ が成り立つ．これは，S_I が 1-MIS であるという仮定に矛盾する． □

最終的な S_O の決定について述べる．$S_A(S_I)$ を，隣接する勝者が 2 つ未満であるような S_I に属す全ての頂点からなる集合とし，以降，単に S_A と表す．$S_B(S_I)$ を，$v.\texttt{parent} \notin S_A$ であるような全ての勝者 v からなる集合とし，以降，単に S_B と表す．これらの定義において，$S_A \cup S_B$ は独立集合であり，S_I が 1-MIS でないならば，$|S_A \cup S_B| \geq |S_I| + |S_B|/2 \geq |S_I| + 1$ が成り立つ．これは，次の (i)，(ii) が成り立つからである．(i) 補題 4.3 より，$S_B \neq \emptyset$ が成り立つ．(ii)$S_I \setminus S_A$ に属す各頂点は，その隣接頂点に少なくとも 2 つの勝者が存在する．したがって，$|S_B| \geq 2|S_I \setminus S_A|$ が成り立つ．一方で，S_I が 1-MIS ならば，補題 4.4 より，$S_A = S_I$ と $S_B = \emptyset$ が成り立つ．ここで，S_I が 1-MIS でないならば，$S_A \cup S_B$ は独立集合であるが，極大独立集合であることは保証されていない．図 4 の例では，$S_A \cup S_B = \{11, 31, 53, 62, 71\}$ であり，$S_A \cup S_B \cup \{28\}$ も独立集合となるため，$S_A \cup S_B$ は極大独立集合ではない．そこで，$S_C(S_I)$ を，次の 2 つを満たすような非勝者 $p_i \notin S_I$ の極大集合とし，以降，単に S_C と表す．

- p_i の隣接頂点に $S_A \cup S_B$ に属す頂点が存在しない．
- S_C に属し，p_i より識別子が小さいような隣接頂点が存在しない．

集合 $S_A \cup S_B \cup S_C$ は，$p_i \in S_C$ に属す各頂点が $S_A \cup S_B \cup S_C$ に属す隣接頂点を持たないため，独立集合である．さらに，$S_A \cup S_B \cup S_C$ が極大独立集合である．なぜならば，もし，$S_A \cup S_B \cup S_C$ が極大独立集合でないならば，$S_A \cup S_B \cup S_C$ に属す隣接頂点を持たないような，非勝者 $v \in V \setminus S_I$ が存在するはずである．しかし，そのような非勝者 v は，S_C の定義より，S_C に属すはずである．よって矛盾が生じるので，$S_A \cup S_B \cup S_C$ は極大独立集合である．以上より次の 3 つの補題が成り立つ．

補題 4.5. S_I が極大独立集合ならば，$S_A \cup S_B \cup S_C$ は極大独立集合である．

補題 4.6. S_I が極大独立集合であり，1-極大独立集合でないならば，$|S_A \cup S_B \cup S_C| \geq |S_I| + |S_B|/2 + |S_C| \geq |S_I| + 1$ が成り立つ．

補題 4.7. S_I が 1-極大独立集合ならば，$S_A \cup S_B \cup S_C = S_I$ と $S_B = S_C = \emptyset$ が成り立つ．

したがって，$S_O = S_A \cup S_B \cup S_C$ である．S_I が 1-極大独立集合でないならば，$|S_O| \geq |S_I| + 1$ が成り立ち，S_I が 1-極大独立集合であるならば，$S_O = S_I$ が成り立つ．図 4 において，$S_A = \{71\}$,，$S_B = \{11, 31, 53, 62\}$,，$S_C = \{28\}$

であり，$|S_O| = 6 > 3 = |S_I|$ が成り立つ．図 3 は，S_I が図 2 で示されるような Inc の 1 回の実行により導かれる S_O を表す．

証明は割愛するが，上記補題により，次の 3 つの補題が得られ，定理 4.11 が導ける．

補題 4.8 (Shiftable Convergence)．状況 $\mathcal{L}_{\text{input}}$ から始まる Inc の任意の極大実行 ϱ は，$\gamma^{\text{copy}} \in \mathcal{L}_{\text{input}}$ であるような状況 γ で終了する．

補題 4.9 (Loop Convergence)．$\varrho_0, \varrho_1 \ldots$ を Inc の極大実行の無限な系列とする．ここで，各 $i \geq 0$ に対して，$\varrho_i = \gamma_{i,0}, \gamma_{i,1}, \ldots, \gamma_{i,s_i}$，$\gamma_{0,0} \in \mathcal{L}_{\text{input}}$，$\gamma_{i+1,0} = \gamma_{i,s_i}^{\text{copy}}$ が成り立つとする．そのとき，ある $j \leq n$ に対して，$\gamma_{j,s_j} \in \mathcal{C}_{\text{goal}}(Inc, E_{\text{MIS}})$ と $R(\varrho_0) + R(\varrho_1) + \ldots R(\varrho_j) = O(n)$ が成り立つ．

補題 4.10 (Correctness)．任意の状況 $\gamma \in \mathcal{C}_{\text{goal}}(Inc, E_{\text{MIS}})$ は $\mathcal{L}_{\text{1MIS}}$ を満たす．

定理 4.11. アルゴリズム $\mathbf{Loop}(Inc, E_{\text{MIS}}, Init)$ は，述語 $\mathcal{L}_{\text{1MIS}}$ に対するサイレントな自己安定アルゴリズムである．任意の状況から始まる $\mathbf{Loop}(Inc, E_{\text{MIS}}, Init)$ の極大実行は $O(nD)$ ラウンド以内に終了する．アルゴリズム $\mathbf{Loop}(Inc, E_{\text{MIS}}, Init)$ の 1 頂点あたりの空間計算量は $O(\log n)$ ビットである．任意の状況 $\gamma \in \mathcal{C}_{\text{goal}}(Inc, E)$ は $\mathcal{L}$ を満たす．

5 結論

反復合成 [3] を用いて，1-極大独立集合問題を解く，サイレントな自己安定アルゴリズムを提案した．n をグラフの頂点数，D をグラフの直径とすると，収束時間は $O(nD)$ ラウンドであり，1 頂点あたりの空間計算量は $O(\log n)$ ビットである．

謝辞 本研究は，JSPS 科研費 17K19977, 18K18000, 19H04085,19K11826，および JST SICROP，JPMJSC1606 の支援を受けたものです．

参考文献

[1] Baruch Awerbuch, Michael Luby, Andrew V. Goldberg, and Serge A. Plotkin. Network decomposition and locality in distributed computation. In *30th Annual Symposium on Foundations of Computer Science*, pages 364–369. IEEE, 1989.

[2] Béla Bollobás, Ernest J. Cockayne, and Christina M. Mynhardt. On generalised minimal domination parameters for paths. In *Annals of Discrete Mathematics*, volume 48, pages 89–97. Elsevier, 1991.

[3] Ajoy K. Datta, Laurence L. Larmore, Toshimitsu Masuzawa, and Yuichi Sudo. A self-stabilizing minimal k-grouping algorithm. In *Proceedings of the 18th International Conference on Distributed Computing and Networking*, page 3. ACM, 2017.

[4] Ajoy K. Datta, Laurence L. Larmore, Toshimitsu Masuzawa, and Yuichi Sudo. A self-stabilizing minimal k-grouping algorithm. *arXiv preprint arXiv:1907.10803*, 2019.

[5] Edsger W. Dijkstra. Self-stabilizing systems in spite of distributed control. *Communications of the ACM*, 17(11):643–644, 1974.

[6] Shlomi Dolev. *Self-stabilization*. MIT press, 2000.

[7] Michiyo Ikeda, Sayaka Kamei, and Hirotsugu Kakugawa. A space-optimal self-stabilizing algorithm for the maximal independent set problem. In *the Third International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT)*, pages 70–74, 2002.

[8] Eijiro Namba. A hierachical self-stabilizing 1-MIS algorihtm, (in Japanese). Master's thesis, Osaka University, 2017.

[9] Zhengnan Shi, Wayne Goddard, and Stephen T. Hedetniemi. An anonymous self-stabilizing algorithm for 1-maximal independent set in trees. *Information Processing Letters*, 91(2):77–83, 2004.

[10] Sandeep K. Shukla, Daniel J. Rosenkrantz, S. Sekharipuram Ravi, et al. Observations on self-stabilizing graph algorithms for anonymous networks. In *Proceedings of the second workshop on self-stabilizing systems*, volume 7, page 15, 1995.

[11] Volker Turau. Linear self-stabilizing algorithms for the independent and dominating set problems using an unfair distributed scheduler. *Information Processing Letters*, 103(3):88–93, 2007.