リングネットワークにおける一様な自己安定 k-相互排除システム

角川裕次† 山下雅史†

分散システムにおけるフォールトトレランスを実現するために，自己安定の概念が1974年 Dijkstra によって提案された．自己安定システムとは，システムの初期状況に関係なく，有限時間内にシステムが正しい（安定な）状況に遷移するシステムである．すなわち，自己安定システムは，一過性のエラーが生じても有限時間内にシステムが（自動的に）再び正しく動作するシステムである．このため，自己安定の概念はフォールトトレランスの理論的基盤と考えられる．分散 k-相互排除問題は，同時に臨界領域に存在するプロセスの数が k 以下であることを保証する問題である．本論文では，自己安定 k-相互排除問題を考え，自己安定 k-相互排除システムを提案し，その正当性を示す．提案するシステムは，一様なリングネットワーク上で実行される．本稿では単方向リングネットワークと双方向リングネットワークを検討する．さらに，任意の k 個のプロセスが同時に臨界領域に入ることができる，という自然な要請を加えた自己安定 k-相互排除問題は単方向リングでは解くことができないことも示す．われわれの提案する双方向リングネットワーク上の自己安定 k-相互排除システムはこの要請を満たしている．

Uniform Self-Stabilizing k-Mutual Exclusion Systems on Ring Networks

HIROTSUGU KAKUGAWA† and MASAFUMI YAMASHITA†

The concept of *self-stabilization* is introduced by Dijkstra to design fault tolerant distributed systems. A self-stabilizing system is a system such that the system converges to a stable (legitimate) state within a finite time regardless of its initial state; that is, even if any transient failure happens when it is working, it will resume a correct system state in a finite time. The k-mutual exclusion problem is the problem of guaranteeing that at most k processes can enter their critical sections. This paper discusses the self-stabilizing k-mutual exclusion problem on bidirectional and unidirectional rings, and proposes uniform self-stabilizing k-mutual exclusion systems (algorithms). The systems proposed are uniform in the sense that all processes are identical. We also show that if we further impose the following natural requirement on systems, then there are no self-stabilizing k-mutual exclusion systems on unidirectional rings: any k processes can enter critical sections at the same time. The systems we propose for bidirectional rings satisfy this requirement.

1. まえがき

耐故障性の高い分散システムを実現するために，自己安定の概念が1974年 Dijkstra によって提案された[3]．自己安定システムとは，システムの初期状況に関係なく，有限時間内にシステムが正しい（安定な）状況に遷移するシステムである．すなわち，自己安定システムは以下の望ましい性質を持っている[20]：

- システムの初期化を行う必要がない．
- 一過性の障害（transient failure）から有限時間内に回復ができる．

† 広島大学工学部第二類
Department of Electrical Engineering, Hiroshima University

このため，自己安定の概念はフォールトトレランスの理論的基盤と考えられ，近年活発な研究がなされてきている．

自己安定の概念，モデル化，およびそのアルゴリズムについては，増澤と片山が素晴らしい解説を「情報処理」誌によせているので詳細はそちらに譲ることにし[16]，ここでは本論文に関係のある相互排除の部分について文献16)を引用しながら当該研究の現状説明を試み，あわせてわれわれの対象とする k-相互排除問題を紹介するとともにその研究目的について述べる．

n 個のプロセスをリング上に通信リングで結合した分散システムが本研究の対象である．プロセス間の通信の方式には，(1)隣接するプロセスの状態を知ることのできる**状態通信モデル**，(2)各リングに対して置

かれた共有レジスタを介して情報を交換する**レジスタ通信モデル**，そして(3)メッセージの送受信によって情報を交換する**メッセージ通信モデル**，の三つが考えられているが，本研究では最もよく研究されてきた状態通信モデルを採用する．

一般に，分散システムに属するいくつかのプロセスが情報を交換し同期をとりながら実行を並行に進めることによって分散システムは進む．プロセスが実行を進める順序をスケジュールと呼ぶ．したがって，スケジュールはプロセスの部分集合の無限列である．任意のスケジュールが許されるモデルをDデーモン (Distributed daemon) と呼ぶ．逆に，各時点でただ一つのプロセスしか実行されないモデルをCデーモン (Central daemon) という．これは，マルチプログラミング環境で動作する単一 CPU の計算機に実現された分散システムのモデル化であり，われわれはこのモデルを採用する（ほかに R/W デーモン (Read/Write daemon) があるが，本論文に直接関係ないので省略する．詳しくは文献16)を参照されたい)．

本論文が検討する問題，k-相互排除問題 (k-mutual exclusion problem) は，最大 k 個のプロセスが臨界領域に進入できることを保証する問題であり，基本的な分散問題の一つに挙げられている[13]．$k=1$ の場合が，いわゆる相互排除問題であり，多くの自己安定アルゴリズムの研究が従来から続けられている[2),3),5)~9),17)]．本論文では，従来の方針にのっとり，この問題を仮想的な特権トークンがちょうど k 個リングを巡回しており，臨界領域への進入を希望するプロセスは特権トークンの到着を持って臨界領域に進入すると考え，「臨界領域に進入できる特権を持つプロセスが“ちょうど k 個である”ことを保証する問題」と見なす．より厳密には，(1)特権を持つプロセスだけがその実行によって分散システムの状態を変化させることができ，(2)すべてのプロセスは有限時間内に特権を手に入れることができなければならない，という条件のもとで，分散システムがちょうど k 個の特権を持つことを保証する問題を考える．

われわれはCデーモンを採用するが，このことで問題が自明とはならないことに注意せよ．われわれの目的は，特権を持つプロセスの数を k 個に安定させることであって，このことは一時的に実行されるプロセス数が1であることとあまり関係はない（もちろん，Cデーモンを採用する方がDデーモンの場合よりも容易であるのは明らかである)．直観的には，マルチプログラミング環境下での相互排除問題でも簡単に解決できないことを想起すればよいだろう．事実，相互排除問題の場合 ($k=1$) においても，多くの研究がなされてきている．

ネットワークが「一様」であるとは，ネットワーク中のすべてのプロセスが同一である（プロセス識別子すら持たない）場合をいい，「準一様」であるとは，1個（あるいは定数個）の特別なプロセス以外は同一であるときをいう（このほかに，一意なプロセス識別子の付いたモデルも考えられている)．相互排除を行うにはプロセスを一つだけ選び出す必要があるが，ネットワークが一様ならばプロセスは同一なので問題を解くことが難しい．本稿は，一様なネットワークを採用する．

これまでの研究結果を文献16)より以下に引用する*．

1. 準一様な方向感覚付きリングネットワーク上
   - ●文献3)：E. W. Dijkstra
     初めて自己安定アルゴリズムの概念を導入した論文．Cデーモンで相互排除問題を解く，それぞれのプロセスの状態数が K, 4, 3 である三つの自己安定アルゴリズムを示した．
   - ●文献1)：G. M. Brown ら
     Dデーモンの下で，方向感覚付きリングで相互排除問題を解くアルゴリズムを示している．また，Cデーモンの下で動作するアルゴリズムがDデーモンの下でも動作する十分条件として，相互非干渉 (non-interfering) という条件を示した．
2. 一様な方向感覚付きリングネットワーク上
   - ●文献4)：E. W. Dijkstra
     一様なリングネットワークのサイズ（プロセス数）n が合成数ならば，その上での決定性自己安定アルゴリズムが存在しないことを証明している．
   - ●文献2) J. E. Burns ら
     サイズが素数の一様なリングネットワーク上では，Cデーモンで決定性自己安定アルゴリズムが存在することを示した．
   - ●文献7)：T. Herman
     任意のサイズの同期式リングネットワーク上で相互排除問題を解く，コイントスを利用したランダムな自己安定アルゴリズムを示した．リングネットワークのネットワーク状況の対称性を打破するために確

* 文献16) は，「一様」と同じ意味の語として「均一」を，「プロセス」の代わりに「プロセッサ」を用いている．

率が利用されている．

● 文献9)：A. Israeli ら

Dデーモンの下で，一様な任意のネットワークおよびサイズが変化する動的リングネットワーク上で，相互排除問題を解く自己安定アルゴリズムを示した．

3. 方向感覚のないリングネットワーク上

● 文献5)：S. Dolev ら

方向感覚のないリングネットワーク上では，状態通信モデルで相互排除問題を解く自己安定アルゴリズムがないことを証明している．

これらの研究結果はいずれも1-相互排除問題に関する研究である．本稿では自己安定1-相互排除の拡張として，自己安定k-相互排除問題を考察する*．

本論文では，まず，(1)条件 "プロセス数 $n\geq 5$ が素数かつ $3\leq k\leq n-3$" を満たす単方向リングに対する一様な自己安定相互排除システムと，(2)条件 "プロセス数 $n\geq 5$ が素数" を満たす双方向リングに対する一様な自己安定k-相互排除システムを提案し，その正当性を示す．

次に，特権を持っているプロセスのリング上での配置が任意のものから他の任意のものへ遷移することが可能である，という新たな制約を加えた自己安定k-相互排除問題を考案し，この問題は双方向リングに対しては解くことができる場合があるが，単方向リングに対しては条件 "$3\leq k\leq n-3$" を満たすならば解くことができないことを示す．$k\leq 2$ または $k\geq n-2$ の場合には，その制約は有効とならず，制約がない場合に対するシステムを用いることで制約付きの問題を解くことができる．

本論文の構成は以下の通りである．2章で諸定義を行い，3章で Burns と Pachl の自己安定システムを紹介し，4章では本研究で得られた結果を示す．最後に5章で本稿のまとめを行い，今後の課題を述べる．

2. 定　　義

一様なリングシステムは $R=(n, \delta, Q)$ で表される．ここに，n はシステム内のプロセス数，δ は遷移関係，そしてQはプロセスの状態集合である．プロセス $P_0, P_1, \cdots, P_{n-1}$ は，時計回りで配置されている．時計回り方を右とし，その反対を左とする**．Rの**状況** (configuration) γ は各プロセスの状態の組である．すなわち，各プロセス P_i の状態が $q_i\in Q$ であるときの状況は $(q_0, q_1, \cdots, q_{n-1})$ である．Γ をすべての状況よりなる集合，すなわち，$\Gamma=Q_0\times Q_1\times\cdots\times Q_{n-1}$ とする．各 i に対し $Q_i=Q$ であるが，説明の簡単さのためにこの記法を用いる．遷移関係 δ が $Q_i\times Q_{i-1 \bmod n}\times Q_i$ の部分集合 ($Q_i\times Q_{i-1 \bmod n}\times Q_{i+1 \bmod n}\times Q_i$ の部分集合) であるとき，R は一様な単方向リングシステム (一様な双方向リングシステム) と呼ばれる．プロセス番号に関する演算は n を法として行い，以降では mod$n$ は省略する．以下では，一様な単方向リングシステム (一様な双方向リングシステム) を単に単方向リング (双方向リング) と呼ぶ．

$q_i\in Q(0\leq i<n)$ を P_i の状態とする．単方向リング (双方向リング) において，ある $q\in Q_i$ に対し $(q_i, q_{i-1}, q)\in\delta((q_i, q_{i-1}, q_{i+1}, q)\in\delta)$ が成立するときかつそのときのみに限り P_i が**特権を持つ**という．すなわち P_i が特権を持つか否かは隣接プロセスとそれ自身の状態に依存する．P_i が特権を持つときのみに限って P_i は実行 (動作) 可能である．P_i が**動作を行う**とは P_i の状態が先述の特権保持の条件におけるある状態 $q\in Q_i$ のうちいずれかに状態遷移を行うことである．これにより R の状況は $\gamma'=(q_0, q_1, \cdots, q_{i-1}, q, q_{i+1}, \cdots, q_{n-1})$ に遷移する．この状況の遷移関係を $\gamma\rightarrow\gamma'$ と書き，$\rightarrow$の反対推移閉包を$\rightarrow^*$と書く．

一般に，特権を持つプロセスは二つ以上存在するが，特権を持っているプロセスのうちいずれか一つが選ばれて実行される．これを**Cデーモン** (Central daemon) と呼ばれるモデルである．なお，本稿では，特権を持つプロセスは有限時間内に実行される，という条件を加える*．この条件をCデーモンの**公平性**と呼ぶ．

初期状況 $\gamma_0\in\Gamma$ より始まる R の**計算**または**遷移列** Λ は，$\gamma_0, \gamma_1, \gamma_2, \cdots (\gamma_j\rightarrow\gamma_{j+1}, j\geq 0)$ なる (無限) 列である．

$\Lambda\subseteq\Gamma$ を R の状況の集合とする．次の条件が成り立つときかつそのときのみに限り R は Λ に関して**自己安定** (self-stabilizing) であるといわれる：

デッドロックフリー　各状況 $\gamma\in\Gamma$ に対し $\gamma\rightarrow\gamma'$ である少なくとも一つの $\gamma'\in\Gamma$ が存在する．

閉包性　任意の $\gamma\in\Lambda$ と $\gamma'\in\Gamma$ に対し $\gamma\rightarrow\gamma'$ ならば $\gamma'\in\Lambda$ である．

---

* 近年，(自己安定ではないが) 分散k-相互排除問題が研究されつつある[10),11),15),18),19),21)]．

** どのプロセスも大局的に矛盾のない右と左を知っている方向感覚付きのリングネットワークを本稿では取り扱う．

* この条件がないと，自己安定k-相互排除アルゴリズムは存在しないことが容易に示される．

ライブロックフリー 任意の $\gamma_0 \in \Gamma$ と任意の（無限）遷移列 $\Delta=\gamma_0, \gamma_1, \cdots$ に対し，$\gamma_j \in \Lambda$ である j が存在する．

公平性 任意の P_i $(0 \leq i < n)$ と Λ に属する状況よりなる任意の（無限の）遷移列 $\Delta=\lambda_0, \lambda_1, \cdots$ に対し，P_i による遷移が無限回存在する．

なお，$\lambda \in \Lambda$ は**正しい状況** (a legitimate configuration) と呼ばれ，Λ は**正しい状況の集合** (a set of legitimate configurations) と呼ばれる．一様なシステム $R=(n, \delta, Q)$ が $\Lambda \subseteq \Gamma$ に関して自己安定であるとき，$S=(n, \delta, Q, \Lambda)$ は**一様な自己安定システム** (a uniform self-stabilizing system) であるという．

では，本稿で議論する二つの型の自己安定 k-相互排除問題を定義する．

【定義1】 1型自己安定 k-相互排除問題は，一様なリングシステム $S=(n, \delta, Q, \Lambda)$ を定める問題である．なお，Λ は以下の条件を満たさなくてはならない．(1) $R=(n, \delta, Q)$ は Λ に対して自己安定である．(2) 各 $\lambda \in \Lambda$ において，ちょうど k 個のプロセスが特権を持っている． □

【定義2】 2型自己安定 k-相互排除問題は，定義1の条件に加えて，以下の条件を満たす一様なリングシステム $S=(n, \delta, Q, \Lambda)$ を定める問題である．$\Pi(\lambda)$ を状況 λ で特権を持つプロセスの集合とする．U を k 個の異なるプロセスよりなる任意の集合とする．このとき，各 $\lambda \in \Lambda$ に対し，計算 $\lambda \rightarrow^* \lambda'$ が存在して $\Pi(\lambda')=U$ となる． □

2型自己安定 k-相互排除問題では，単にシステム内に k 個の特権をおくだけでなく，任意の特権の配置から他の任意の配置への遷移を可能とする計算が存在しなければならない，という制約が加わっている．一方，1型自己安定 k-相互排除問題ではそのような制約はなく，ある限られた特権の配置しか生じなくても構わない．問題の定義より，2型自己安定 k-相互排除問題を解くシステムは1型自己安定 k-相互排除問題も解く．後の証明よりわかるが，単方向リングでのいかなるシステムでも，安定状況ではある限られた特権の配置しか起こらない．このようなことが生じないことを保証するのが2型問題であり，自由な特権の移動が保証される．

3. Burns と Pachl のシステム

4章で提案する自己安定 k-相互排除システムは，Burns と Pachl によって文献2)で提案された一様な単方向リング上で動作する自己安定1-相互排除システムを基にして設計する．われわれの提案するシステムの正当性の証明を簡明にするために，本章で Burns と Pachl の単方向リングでの一様な自己安定 1-相互排除システム $S_0=(n, \delta_0, Q_0, \Lambda_0)$ を示す．以降，Burns と Pachl の自己安定システムのことを BP と書く．

プロセス数 $n \geq 5$ は素数とする．各プロセス P_i の状態 $q_i \in Q_0$ は $l_i.t_i$ なる二つの成分よりなる．$l_i \in \{0, 1, \cdots, n-2\}$，$t_i \in \{0\} \cup \{2, 3, \cdots, n-2\}$ を，それぞれラベル部，タグ部と呼ぶ．

準備として，以下を定める．

$$R_A(i) \equiv (l_i \neq l_{i-1}+1) \wedge (l_i \neq 0 \vee t_{i-1}=0 \vee t_{i-1} \neq l_i - l_{i-1} \vee t_{i-1} < t_i),$$

$$R_B(i) \equiv (l_i = l_{i-1}+1) \wedge (t_{i-1} \neq t_i) \wedge (l_i \neq 0)$$

S_0 の遷移関係 δ_0 は以下の通りである．なお，遷移関係は読みやすさのために，以下の規則の集合の形で与える．

規則 BP-A：

If $R_A(i)$ then $l_i.t_i := (l_{i-1}+1).(l_i - l_{i-1})$.

規則 BP-B：

If $R_B(i)$ then $l_i.t_i := l_i.t_{i-1}$.

なお，ラベルとタグに対する演算は，$n-1$ を法として行われる．

正しい状況は以下のものである（なお l は任意のラベル値であり，下線を引いた状態を持つプロセスが特権を持つ）．

$$\cdots, l-2.0, l-1.0, \underline{l.0}, l.0, \underline{l+1.0}, l+2.0, \cdots$$

なお，特権を持つプロセスが規則を実行すると，この状況は

$$\cdots, l-2.0, l-1.0, l.0, l+1.0, \underline{l+1.0}, l+2.0, \cdots$$

となり，特権は右隣のプロセスに移る．

S_0 に対し次の補題が成立する．

【補題1[2]】 デッドロックフリー

任意の状況 γ に対しあるプロセス P が存在し，P は規則 BP-A または BP-B による特権を持つ． □

このシステムで用いられるいくつかの用語を定義しておく．連続したプロセス P_1, P_2 の状態をそれぞれ $l_1.t_1$, $l_2.t_2$ とするとき，$l_2 \neq l_1+1 (\mathrm{mod}\, n-1)$ ならば，P_2 に**ギャップ**があるといい，その**ギャップサイズ**は $l_2 - l_1$ である．**セグメント**とは，ギャップを含まない連続したプロセスの極大部分列 $s=(P_i, P_{i+1}, \cdots, P_j)$ をいい，$P_i(P_j)$ を s の**先頭**（**末尾**）プロセスと呼ぶ．セグメント $s=(P_i, P_{i+1}, \cdots, P_j)$ において，すべての x $(i \leq x \leq j)$ に対し $t_x = l_{j+1} - l_j (\mathrm{mod}\, n-1)$ であると

き，セグメント s は well formed であると呼ばれる．

BP システムが安定してゆく様子の概略を説明する．任意の状態では，(高々 n の) 複数のセグメントが存在する．正常な状況は，セグメント数が1であり，しかもそれは well formed である．規則 BP-A, BP-B どちらを適用しても，セグメントの数は増加しない (規則 BP-A は特権の移動ならびにセグメント数の減少の働きをし，規則 BP-B はセグメントを well formed にする働きをする)．セグメントの数を一定に保とうとしても，すべてのセグメントが有限時間内に well formed となり，それに伴って必ずセグメント数が減少する．ひとたびセグメント数が1になると，有限時間内にそのセグメントは well formed となる．このようにして，システムが安定してゆく．

4. 結　　果

4.1 単方向リング

単方向リングでの1型自己安定 k-相互排除問題を解く自己安定システム $S_1=(n, \delta_1, Q_1, \Lambda_1)$ を以下に示す．なお $n\geq 5$ は素数とする．

状態集合を $Q_1=\{l.t\}, l\in\{0,1,\cdots,n-2\}$, $t\in\{0\}\cup\{2,3,\cdots,n-2\}$ とする．l, t はそれぞれラベル，タグと呼ばれる．遷移関係 δ_1 は，読みやすさのために，規則の集合の形で与える．なお，以下の記述では各プロセス P_i ごとにアルゴリズムを与えているが，すべてのプロセスに対し同一のアルゴリズムであることに注意されたい．($R_A(i), R_B(i)$ は，BP の定義で示したものとする．)

規則 Uni-A :

If $R_A(i)$ then

$l_i.t_i := (l_{i-1}+1).(l_i - l_{i-1})$.

規則 Uni-B :

If $R_B(i)$ then

$l_i.t_i := l_i.t_{i-1}$.

規則 Uni-C :

If $(n-k\leq l_i\leq n-2)\wedge\neg(R_A(i)\vee R_B(i))$ then

何もしない．

なお，ラベルとタグに対する演算は，$n-1$ を法として行われる．

正しい状況の集合 Λ_1 は以下の形をした状況の集合である．

$\cdots, l-2.0, l-1.0, l.0, l.0, l+1.0, l+2.0, \cdots$,

ここで l は任意の値のラベルを表す． □

では，このアルゴリズムの正当性を証明する．

【補題2】 正しい状況においてちょうど k 個のプロセスが特権を持つ．また，閉包性が成立する．

(証明) 任意の正しい状況 $\lambda\in\Lambda_1$ を

$\cdots, l-2.0, l-1.0, l.0, l.0, l+1.0, l+2.0, \cdots$

とおく．λ において規則 Uni-A による特権を持つプロセスを P_0 とし，P_0 の状態を $l_0.t_0$ とする．(なお，$t_0=0$ である．)

- $l_0\in\{n-k,\cdots,n-2\}$ のとき．各 $l\in\{n-k,\cdots, l_0-1, l_0+1,\cdots,n-2\}$ に対し $l_i=l$ なる P_i はちょうど一つ存在し，それらはいずれも規則 Uni-C のみの特権をもつ．また，$l_i=l_0$ なる P_i はちょうど二つ存在するが，このうちの一つは規則 Uni-C による特権，他方 (P_0) は規則 Uni-A による特権を持つ．ゆえに，規則 Uni-C による特権を持つプロセスは $k-1$ 個あり，これらはいずれも P_0 ではない．よって，全体で k 個のプロセスが特権を持つ．次に閉包性を示す．規則 Uni-C を適用しても状況は変わらないので，P_0 が規則 Uni-A による特権を適用した後の状況 γ' を考える．γ' は

  $\cdots, l-2.0, l-1.0, l.0, l+1.0, l+1.0, l+2.0, \cdots$

  であり，これは正しい状況である．
- $l_0\notin\{n-k,\cdots,n-2\}$ のとき．各 $l\in\{n-k,\cdots,n-2\}$ に対し $l_i=l$ なる P_i はちょうど一つ存在する．ゆえに，規則 Uni-C 特権を持つプロセスは $k-1$ 個あり，それらはいずれも P_0 ではない．よって，全体で k 個のプロセスが特権を持つ．閉包性は上の場合と同様にして示される．

□

【補題3】 公 平 性

(証明) BP と S_1 の定義より，任意の $\lambda\in\Lambda_1$ において，規則 Uni-A による特権を持つプロセス P が存在する．Cデーモンの公平性より，有限時間内に P が実行され，規則 Uni-A による特権を持つプロセスは一つ右に移動する． □

【補題4】 S_1 はデッドロックフリーである．

(証明) 規則 Uni-A, Uni-B の条件は BP のものと同一である．ゆえに，文献2)での BP のデッドロックフリー性の証明と同一の証明により，任意の状況 γ において，少なくとも一つのプロセスで規則 Uni-A または Uni-B のどちらか一方の条件が成立する．よってデッドロックは生じない． □

【定理1】 S_1 は1型自己安定 k-相互排除問題を解く．

(証明) 規則 Uni-C は適用されてもプロセスの状態を変えないので規則 Uni-C の適用は除外して考える．規則 Uni-A, Uni-B の条件は BP のものと同一である．ゆえに，文献2)での BP の正当性の証明と同一の証明により，任意の状況 γ_0 から有限時間内に以下の状況（すなわち，BP の意味で正しい状況）γ_1 に遷移する：

$$\cdots, l-2.0, l-1.0, l.0, l.0, l+1.0, l+2.0, \cdots$$

(なお，補題4の証明より，規則 Uni-C だけの特権しか存在しないような状況は存在しないことに注意せよ．)これは正しい状況である． □

【定理2】 プロセス数 $n\geq 6$ と $k, (3\leq k\leq n-3)$ に対し，単方向リング上で2型 k-相互排除問題を解く自己安定システムは存在しない．

(証明) 単方向リング上で2型 k-相互排除問題を解く自己安定システムが存在すると仮定する．Λ を正しい状況の集合とする．任意の正しい状況を $\gamma_0\in\Lambda$ とすると，仮定によりある状況 $\gamma_1\in\Lambda$ とある計算 $\gamma_0\rightarrow^*\gamma_1$ が存在し，連続した k 個のプロセスが γ_1 において特権を持つ．一般性を失うことなく，$P_0, P_1, \cdots, P_{k-1}$ が γ_1 において特権を持つと仮定する．$P_i\ (0\leq i\leq k-2)$ が状態を遷移しても特権の位置が変わらないことは容易にわかる．したがって，特権の移動は，P_{k-1} が何度か状態遷移をした後に P_{k-1} が特権を失い，P_k に特権が移る場合のみに限る．同様にして，P_k が特権を失い，P_{k+1} が特権を持つ．これが繰り返されて，特権は P_{n-1} に移る．(システムが単方向リングなので，特権が P_{n-1} に移るまでは $P_i\ (0\leq i\leq k-2)$ が状態を遷移しても特権の位置が変わらない．)

- $k\leq\lfloor n/2\rfloor$ のとき．k 個の特権が互いに離れて存在するような状況にはならない．
- $k>\lfloor n/2\rfloor$ のとき．特権を持たないプロセスが，互いに離れて存在するような状況にはならない．

□

なお，$n\geq 5$ が素数で，$k=2$, $n-2$, $n-1$ のときには，S_1 を用いて2型 k-相互排除問題を解くことができる．

4.2 双方向リング

次にプロセス数 $n\geq 5$ が素数である双方向リングでの，2型自己安定 k-相互排除問題を解く自己安定システム S_2 を提案する．S_2 は以下の考えに基づいて構成される．k 個の双方向のトラック（リング）より構成される双方向リングを考える．各プロセスは並行して，k 個のトラック各々で（BP を用い）1-相互排除を実行する．S_2 のプロセスが特権を持つのは，そのプロセスが少なくとも一つのトラックで1-相互排除での特権を持つときおよびそのときのみに限る．それぞれのトラックが無限に頻繁に実行されると，各トラックは有限時間内に BP の意味において安定し，各トラックにおける特権の数は1個になる．よって，S_2 における特権の数は高々 k になるが，閉包性を満足するにはどのプロセスも高々一つのトラックでしか1-相互排除での特権を持たないようにしなければならない．

$S_2=(n, \delta_2, Q_2, \Lambda_2)$ を以下に示す．

状態集合を $Q_2=\{(l_j^1.t_j^1, l_j^2.t_j^2, \cdots, l_j^k.t_j^k)\,|\,l_j^i\in\{0,1,\cdots,n-2\}, t_j^i\in\{0\}\cup\{2,3,\cdots,n-1\}\}$ とし，Γ_2 をすべての状況の集合とする．遷移関係 δ_2 は，以下の規則の集合の形で与える．なお，状況を $(q_0, q_1, \cdots, q_{n-1})$, $q_j=(l_j^1.t_j^1, l_j^2.t_j^2, \cdots, l_j^k.t_j^k)$ とする．記述を単純にするために以下の関数と述語を定義する：

$$R_A(i,j)\equiv(l_j^i\neq l_{j-1}^i+1)\wedge(l_j^i\neq 0\vee t_{j-1}^i=0\vee t_{j-1}^i\neq l_j^i-l_{j-1}^i\vee t_{j-1}^i<t_j^i),$$

$$R_B(i,j)\equiv(l_j^i=l_{j-1}^i+1)\wedge(t_{j-1}^i\neq t_j^i)\wedge(l_j^i\neq 0),$$

$$S_p(j)\equiv\{i\,|\,(1\leq i\leq k)\wedge(l_j^i=l_{j-1}^i)\},$$

$$\pi_j\equiv|S_p(j)|, \text{ and}$$

$$R_S(j)\equiv\prod_{1\leq i\leq k}\{((l_j^i=l_{j-1}^i+1)\wedge(l_{j+1}^i=l_j^i))\vee((l_j^i=l_{j-1}^i)\wedge(l_{j+1}^i=l_j^i+1))\vee((l_j^i=l_{j-1}^i+1)\wedge(l_{j+1}^i=l_j^i+1))\}\wedge\forall i, j'(1\leq i\leq k, j-1\leq j'\leq j+1)[t_{j'}^i=0]\wedge\pi_j\geq 1.$$

プロセス P_j の遷移関係 δ_2^j は以下のものとする．記述を簡単にするためにプロセス識別子 $j-1, j, j+1$ を用いているが，すべての j, j' に対して $\delta_2^j=\delta_2^{j'}$ に注意されたい．

規則 Bi-A：

If $\neg R_S(j)\wedge\exists i(1\leq i\leq k)[R_A(i,j)]$ then

$R_A(i', j)$ が真であるすべての i' に対し

$l_j^{i'}.t_j^{i'}:=(l_{j-1}^{i'}+1).(t_j^{i'}-t_{j-1}^{i'})$,

$R_B(i', j)$ が真であるすべての i' に対し

$l_j^{i'}.t_j^{i'}:=l_{j-1}^{i'}.t_{j-1}^{i'}$.

規則 Bi-B：

If $R_S(j)\wedge(\pi_j=1)\wedge(\pi_{j+1}\geq 1)$ then

状態はそのままにする．

規則 Bi-C：

If $R_S(j)\wedge(\pi_j=1)\wedge(\pi_{j+1}=0)$ then

$i'=\min S_p(j)$ に対し

$l_j^{i'}.t_j^{i'} := (l_{j-1}^{i'}+1).(t_j^{i'}-t_{j-1}^{i'})$.

規則 Bi-D:

If $R_S(j) \wedge (\pi_j \geq 2) \wedge (\pi_{j+1} \geq \pi_j - 1)$ then

状態はそのままにする.

規則 Bi-E:

If $R_S(j) \wedge (\pi_j \geq 2) \wedge (\pi_{j+1} < \pi_j - 1)$ then

$i' = \min S_p(j)$ に対し

$l_j^{i'}.t_j^{i'} := (l_{j-1}^{i'}+1).(t_j^{i'}-t_{j-1}^{i'})$.

規則 Bi-F:

If $\exists i (1 \leq i \leq k)[R_B(i,j)]$then

$R_B(i',j)$ が真であるすべての i' に対し

$l_j^{i'}.t_j^{i'} := l_j^{i'}.t_{j-1}^{i'}$.

正しい状況 $\lambda \in \Lambda_2$ は以下のものである：(1)各トラックは BP の意味で正しい状況にあり，(2)各プロセス P_j は高々一つのトラックにおいて 1-相互排除での特権を持っている．正しい状況の集合 Λ_2 は，以下の型をした状況の集合である．$\gamma = (q_0, q_1, \cdots, q_{n-1})$ を $q_j = (l_j^1.t_j^1, l_j^2.t_j^2, \cdots, l_j^k.t_j^k)$ なる状況とするとき，γ に対して以下の条件が成立するときおよびそのときのみに限って $\gamma \in \Lambda_2$ である：

$\forall i, j[t_j^i = 0]$,

各 i に対し $(l_0^i, l_1^i, \cdots, l_{n-1}^i)$ は，ある l^i に対して $(l^i, l^i+1, l^i+2, \cdots, l^i+n-3, l^i+n-2)$ の巡回シフトである（なお，加算は $n-1$ を法とする），

各 j に対し $|\{i | l_j^i = l_{j-1}^i\}| \leq 1$. □

以下に S_2 の正当性を証明する．

【補題5】 正しい状況において，ちょうど k 個のプロセスが特権を持つ．

(証明) 正しい状況の定義より明らか. □

【補題6】 閉 包 性

(証明) 任意の正しい状況を $\lambda \in \Lambda_2$ とする．λ で特権を持つプロセスでは，規則 Bi-B または Bi-C のどちらか一方の条件が成立している（右のプロセスが特権を持っていれば規則 Bi-B で，そうでなければ規則 Bi-C による特権である）．特権を持つプロセス P_j が実行されたとする．

- 規則 Bi-B による特権のとき．実行後の状況は λ と同一である．
- 規則 Bi-C による特権のとき．規則 Bi-C 適用後の状況 γ では P_j は特権を持たず，P_{j+1} はある一つのトラックで規則 BP-A の条件が成立し特権を持つ．この状況は Λ_2 に含まれる. □

【補題7】 公 平 性

(証明) 任意の正しい状況を $\lambda \in \Lambda_2$ とする．$k<n$ なので，特権を持つプロセスの内，右隣のプロセスが特権を持たないようなプロセス $P_a, P_b, \cdots$ が存在する．これらのプロセス $P_a, P_b, \cdots$ は，規則 Bi-C による特権を持っている（これら以外の特権を持つプロセスは規則 Bi-B の条件が成立している）．規則 Bi-B は適用されても状況は変わらず，Cデーモンの公平性により，有限時間内に $P_a, P_b, \cdots$ のいずれかのプロセス P が実行される．P が規則 Bi-C を実行した直後の状況 γ では P は特権を持たず，P の右隣のプロセスが特権を持つ．ゆえに，任意の $\lambda \in \Lambda_2$ より始まる任意の計算に対し，有限時間内にいずれかの特権が右隣のプロセスに移る．したがって，どのプロセスも無限時間内に無限回の特権を持つ. □

【補題8】 デッドロックフリー

(証明) ある状況 $\gamma \in \Gamma_2$ においてデッドロックが生じる，すなわち，すべての規則の条件が偽と仮定する．BP のデッドロックフリー性（補題1）により，各トラックにおいて規則 BP-A または BP-B の少なくとも一方の条件が成立する．もしあるトラックで規則 BP-B の条件が成立すれば，規則 Bi-F による特権が存在し仮定に反する．ゆえに，$\forall i \exists j[R_A(i,j)]$ が成立する．$R_A(i,j)$ の成立する i,j のうちで任意のものを i_0, j_0 とおく．

- $R_S(j_0)$ のとき．$\vee_{r \in \{B,C,D,E\}}$ (規則 Bi-r の条件部) $= R_S(j_0)$ なので，Bi-B, Bi-C, Bi-D, Bi-E のいずれかの条件が真となり矛盾.
- $\neg R_S(j_0)$ のとき．規則 Bi-A の条件が成立するので仮定に反する. □

以下では，S_2 にライブロックが生じないことを示す．

【補題9[14]】 第 i トラックの m 個のギャップのギャップサイズをそれぞれ $g_0, g_1, \cdots, g_{m-1}$ とすれば $\sum_{j=0}^{m-1} g_j = m-1 (\mathrm{mod}\, n-1)$. □

【補題10】 規則 Bi-B および規則 Bi-D 以外の特権の存在しない状況は存在しない．

(証明) 規則 Bi-B かつ/または規則 Bi-D のみの特権しかない状況 γ が存在すると仮定する．BP はライブロックフリーなので，各トラック i で規則 BP-A の条件 $(= R_A(i,j))$ または BP-B の条件 $(= R_B(i,j))$ のいずれかが成立するような j が少なくとも一つ存在する．もしある i,j に対し規則 BP-B の条件が成立すれば，規則 Bi-F の条件が成立し仮定に反する．ゆえに，各トラックにおいて規則 BP-B の条件は成立しない．よって，すべての i,j に対し $\neg R_B(i,j)$ である．

- P_j で規則 Bi-B の条件が成立するとき，$R_S(j)$ が真なので $l_{j+1}^i=l_j^i$ なる i が存在し，かつ，$t_{j+1}^i=t_j^i=0$ である．これより $R_A(i,j+1)$ が真．$\neg R_S(j+1)$ を仮定すれば規則 Bi-A の条件が成立し矛盾．ゆえに $R_S(j+1)$ が真であり，規則 Bi-B, Bi-C, Bi-D, Bi-E のいずれかの条件が成立する．(規則 Bi-B, Bi-C, Bi-D, Bi-E の条件の論理和が $R_S(j)$ であることに注意せよ．) よって，仮定により P_{j+1} は規則 Bi-B または Bi-D による特権を持つ．
- P_j で規則 Bi-D の条件が成立するとき，$\pi_j\geq 1$ なので $l_{j+1}^i=l_j^i$ なる i が存在し，かつ，$t_{j+1}^i=t_j^i=0$ なので $R_A(i,j+1)$ が真．上の場合と同じ理由で P_{j+1} は規則 Bi-B または規則 Bi-D による特権を持つ．

以上の議論により，すべてのプロセスが規則 Bi-B または Bi-D による特権を持つ．よって $\forall i,j[R_S(j)\wedge\neg R_B(i,j)]$ が成立．各トラックでの各ギャップサイズは 0 なので，各トラックでのギャップサイズの総和は 0 である．$R_S(j)$ の定義により，各トラックは高々 $n-1$ 個のセグメントを持つ．このことと補題 9 により，各トラックのセグメント数は 1 である．ゆえに，$\sum_j\pi_j\leq k$ を得る．しかし $R_S(j)$ ならば $\pi_j\geq 1$ なので $\sum_j\pi_j\geq n$ となり矛盾．□

【補題 11】 任意の γ において各第 i トラック $(1\leq i\leq k)$ が BP の意味で正しい状況にあると仮定する．γ_0 より始まる任意の計算列 $\gamma_0\to\gamma_1\to\cdots$ に対しある有限な τ が存在し $\gamma_\tau\in\Lambda_2$ となる．

(証明) 各トラックが BP の意味で正しい状況にあるときの S_2 の振舞いは，以下の (自己安定) システム S_3 の振舞いと等価である．S_3 は n 個のプロセスよりなる双方向リングで，各プロセス P_j は状態 π_j $(0\leq\pi_j\leq k, \sum_{j=0}^{n-1}\pi_j=k<n)$ を持つ*．S_3 は各 π_j が任意の時点で高々 1 になるようにする．安定な状況は，すべての j に対し $0\leq\pi_j\leq 1$ である．なお，$\pi_i\geq 1$ のとき，P_i は**トークン**を持つといい，π_i を P_i の**トークン数**と呼ぶ．遷移関係は以下の通りである：

規則 Bi-B′:

If $(\pi_j=1)\wedge(\pi_{j+1}\geq 1)$ then

なにもしない．

規則 Bi-C′:

If $(\pi_j=1)\wedge(\pi_{j+1}=0)$ then

$\pi_j:=0, \pi_{j+1}:=1$.

規則 Bi-D′:

If $(\pi_j\geq 2)\wedge(\pi_{j+1}\geq\pi_j-1)$ then

なにもしない．

規則 Bi-E′:

If $(\pi_j\geq 2)\wedge(\pi_{j+1}<\pi_j-1)$ then

$\pi_j:=\pi_j-1, \pi_{j+1}:=\pi_{j+1}+1$.

遷移関係の定義より $\sum_{j=0}^{n-1}\pi_j=k$ が常に成立することは容易にわかる．

デッドロックフリー：初期状況では $\sum_{j=0}^{n-1}\pi_j=k$, $2\leq k<n$ であり $\vee_{r\in\{B,C,D,E\}}$ (規則 Bi-r' の条件)$=(\pi_j\geq 1)$ なので，ある j と $r\in\{B',C',D',E'\}$ に対して Bi-r の条件が真となる．次に，条件 Bi-B′ と条件 Bi-D′ のみの特権しか存在しない状況は存在しないことを示す．ある状況において条件 Bi-B′ かつ/または Bi-D′ の特権しか存在しないと仮定する．あるプロセス P_j が条件 Bi-B′ または Bi-D′ による特権を持つとすれば，$\pi_{j+1}\geq 1$. よって P_{j+1} も特権を持つが，仮定によりこの特権は規則 Bi-B′ または Bi-D′ によるものである．この議論を繰り返すことにより，すべてのプロセスが規則 Bi-B′ または Bi-D′ による特権を持つことが導かれるが，これは $\sum_{j=0}^{n-1}\pi_j=k<n$ に矛盾．

閉包性：正しい状況では規則 Bi-B′ または Bi-C′ による特権しか存在せず，これらの規則が適用された後の状況も正しい状況であることは明らか．

ライブロックフリー：ライブロックが生じたと仮定する．デッドロックフリーの証明より，少なくとも一つのプロセスは規則 Bi-C′ または規則 Bi-E′ による特権を持つことがいえる．状況 $\gamma=(q_0,\cdots,q_{n-1})$ に対し $M(\gamma)=\max\{\pi_j|0\leq j<n\}$ とする．$\gamma\to^*\gamma'$ なる任意の γ' に対し $M(\gamma')\leq M(\gamma)$ は明らか．仮定により，ある状況 γ' 以降 $M_0=M(\gamma')\geq 2$ が一定であるような γ' および γ' より始まる計算 Δ が存在する．この Δ において $\gamma'\to^*\gamma''$ なる任意の状況 γ'' が存在し，γ'' 以降は $\pi_j=M_0$ である P_j の数が一定となる．J を，γ'' において $\pi_j=M_0$ かつ $\pi_{j+1}<M_0$ なる j の集合とする．$k<n$ なので，ある i $(0\leq i<n)$ に対し $\pi_i\neq M_0$ である．また $\pi_i=M_0$ である連続したプロセスの内 P_j $(j\in J)$ 以外のプロセスは，規則 Bi-C′ および規則 Bi-E′ による特権を持たない．

---

* S_3は双方向リングであるが，プロセスの次の状態はプロセス自身の状態と右隣のプロセスで定まることに注意せよ．また，P_j が規則を適用すると π_j,π_{j+1} ともに変更を行うため，S_3は厳密にはリングシステムではない．S_3の自己安定性の定義はリングシステムと同様なので，紙面の都合上省略する．

- γ'' 以降の任意の状況において，すべての $j\in J$ に対し P_j が規則 Bi-D′ による特権を持つ場合．γ'' 以降，あるプロセス P_i $(0\leq i<n)$ が規則 Bi-C′ または Bi-E′ を実行したとすると，規則の適用前には $\pi_{i+1}<\pi_i$ が成立している．すなわち，右隣のトークンが少なければ移動が行われている．γ'' 以降は $\pi_i=M_0$ であるプロセス集合は変わらないので，規則 Bi-C′ および Bi-E′ の適用は高々有限回であり，有限時間内に規則 Bi-C′ および Bi-E′ による特権を持つプロセスは存在しなくなり，上記の事実に矛盾する．
- それ以外のとき，すなわち，γ'' 以降のある状況において，ある $j\in J$ に対し P_j が規則 Bi-E′ による特権を持つ場合．このとき $\pi_{j+1}<\pi_j-1=M_0-1$ である．P_j が規則の適用を行わない限りは π_{j+1} は増加しないので，P_j が規則 Bi-E′ による特権をひとたび持つと，その特権は P_{j+1} の実行によって失われない．したがって，Cデーモンの公平性により有限時間内に P_j が規則 Bi-E′ を適用する．これに伴い，P_j のトークン数は $\pi_j-1=M_0-1$ となり，P_{j+1} のそれは $\pi_{j+1}+1<M_0$ となる．他のプロセスのトークン数はそのままであるので，M_0 個のトークンを持つプロセスの数が減少したことになり矛盾．

よってライブロックは生じず，S_3 システムは有限時間内に正しい状況に遷移する． □

【補題 12】 S_2 はライブロックフリーである．

（証明） 補題 10 により，状況が永遠に変化しないライブロックは生じない．ゆえに，任意の状況 $\gamma\in\Gamma_2$ において，規則 Bi-A，Bi-C，Bi-E，Bi-F のいずれかによる特権を持つプロセスが少なくとも一つ存在する．Cデーモンの公平性によりそれらプロセスの内いずれかが有限時間内に実行され，別の状況 $\gamma'\neq\gamma$ に遷移する．よって，以下では規則 Bi-B，Bi-D の実行は除外して考える．

ある状況 $\gamma\in\Gamma_2$ とある実行列 Δ が存在し，有限時間内に正しい状況に遷移しないと仮定する．有限時間内にすべてのトラックが BP の意味で正しい状況となれば，補題 11 により有限時間内に S_2 は正しい状況に遷移する．したがって，Δ において $\gamma\rightarrow^*\gamma'$ なる γ' と，あるトラック I が存在し，$\gamma'\rightarrow^*\gamma''$ なるすべての γ'' において第 I トラックは BP の意味で正しい状況になく，かつ，γ'' 以降変化が生じないと仮定できる（なぜなら，変化が生じれば BP の意味で正しい状況になるからである）．

S_2 の定義より，あるプロセス P が規則 Bi-A，Bi-C，Bi-E を適用することはあるトラックに対して BP の規則 BP-A を（および，場合により別のトラックに対して規則 BP-B を）適用することであり，規則 Bi-F を適用することは，あるトラックに対して規則 BP-B を適用することに注意されたい．

規則 Bi-A，Bi-C，Bi-E の内少なくとも一つは Δ において無限回適用されるので，無限回状況の変化するトラックが存在し，有限時間内に BP の意味で正しい状況に遷移する．そのようなトラックの内で番号のいちばん小さなものを I_0 とする．第 I_0 トラックでは，規則 BP-A による特権が左から右へ順々に移動して行く．このとき，第 I トラックのプロセス P に規則 BP-B による特権があると，第 I_0 トラックで P が規則 BP-A の実行の際に第 I トラックで規則 BP-B が適用され仮定に反する．ゆえに，第 I トラックには規則 BP-B による特権は存在せず，規則 BP-A による特権しか存在しない．第 I トラックにおいて規則 BP-A による特権を持つプロセスの添字の集合を J とする．すると，各 $j\in J$ に対し P_j が第 I_0 トラックにて規則 BP-A による特権を持つときには P_j は規則 Bi-E による特権を持つ．なぜなら，このとき，P_j の特権が規則 Bi-A または Bi-C によるものであれば，規則の適用時に第 I トラックで規則 BP-A が適用され仮定に反するからである（第 I_0 トラックは BP の意味で正しい状況にあり，規則 BP-A による特権が無限にリングを巡回しているので，P_j に規則 BP-A による特権が巡ってきたときには，P_j は規則 Bi-B，Bi-D 以外による特権を持つ）．また，規則 Bi-E の定義により $I_0<I$ である．P_j で $R_S(j)$ が成立するので，第 I トラックのセグメント数は高々 $n-1$ である．各 $j\in J$ に対し $t^I_{j-1}=t^I_j=t^I_{j+1}=0$ であり，かつ第 I トラックに規則 BP-B による特権がないので，もし第 I トラックでのセグメント数が 1 と仮定すれば第 I トラックのタグがすべて 0，すなわちそのトラックは BP の意味で正しい状況となり矛盾．ゆえに，第 I トラックのセグメント数 s は $2\leq s\leq n-1$ である．また，第 I トラックで規則 BP-A による特権を持つ各 P_j，$j\in J$ に対し，P_j でのギャップサイズは 0 である．

$2\leq s\leq n-1$ なので，補題 9 によりギャップサイズが 0 以外のギャップが存在する．すなわち，γ での第 I トラックのセグメント数を s，ギャップサイズの列を時計回りに $g_0, g_1, \cdots, g_{s-1}$，とすれば，ある h が存

在し $g_h \neq 0$ となる．P_H にギャップ g_h があるとすると，P_H は規則 BP-A による特権を持たない．なぜなら，第 I_0 トラックの規則 BP-A による特権が P_H に移ってきた状況において，もし第 I トラックに規則 BP-A による特権があればギャップサイズが 0 以外なので $R_S(H)$ が偽となり，規則 Bi-A の適用が行われ第 I トラックに変化が生じるからである．よって P_H では $\neg R_A(I,H)=(l_H^I=0)\wedge(t_{H-1}^I\neq 0)\wedge(l_{H-1}^I\neq 0)\wedge(t_{H-1}^I\geq t_H^I)$ が成立する．よって，ギャップサイズが 0 以外のセグメントの左端のプロセスのラベルは 0 である．

セグメント S を，ギャップサイズが 0 以外であり，かつ，S の左隣のセグメント S_{-1} のギャップサイズが 0 であるものとする（必ずそのようなものが存在することは明らか）．S の先頭のプロセス P_0 のラベルの値は 0 であり，そのプロセスの左のプロセス P_{-1}(S_{-1} の末尾プロセス）のラベルは 0 以外で，かつ，タグの値は 0 以外である．よって，規則 BP-B がすべてのプロセスで不成立なので，0 をラベルに持つプロセスが S_{-1} に含まれ，S_{-1} の長さは 2 以上である．第 I トラックでの各プロセスのラベルを，P_H を先頭にして書けば，

$$l_{1,1}, l_{1,2}, l_{1,3}, \cdots, l_{2,1}, l_{2,2}, l_{2,3}, \cdots, l_{3,1}, l_{3,2}, l_{3,3}, \cdots$$

となる（ただし，すべての i,j に対し $l_{i,1}=0$ かつ $l_{i,j}\leq l_{i,j+1}$）．すなわち，第 I トラックのラベル列は 0 より始まる非減少列の並びとなっている．S_{-1} の先頭のプロセス P_L のラベルが 0 と仮定する．S_{-1} のギャップサイズが 0 なので規則 BP-A の条件が成立．ゆえに $L\in J$ であり P_L のタグは 0．どのプロセスも規則 BP-B による特権を持たないことと，セグメント数が 2 以上なので一つのセグメント内にラベル 0 を持つプロセスが高々一つであることより，S_{-1} のプロセスはいずれも 0 をタグに持ち，矛盾．よって，P_L のラベルは 0．しかし，セグメント数が 2 以上なので，これを満たす非減少列の並びは存在せず，矛盾．

よって，各トラックは有限時間内に正しい状況に遷移し，補題 11 により本補題が成立する． □

以上より，次の定理が成立する．

【定理 3】 S_2 は 2 型 k-相互排除問題を解く自己安定システムである． □

5. むすび

本稿では，C デーモンの下での一様なリングネットワークでの k-相互排除問題について議論した．双方向リングでの，任意の特権の配置から他の任意の配置に遷移することのできる自己安定システムを提案し，その正当性を示した．一方，単方向リングでは，特権の配置に関して条件のない自己安定システムは存在するが，任意の特権の配置に遷移できる自己安定システムは存在しないことを示した．本稿で示した自己安定システムはプロセス数 n が素数の場合であった．n が合成数で k を因数に持たない場合は，k-相互排除問題を解く一様な自己安定システムが存在しないことは容易に示すことができる．それ以外の場合，すなわち n が k を因数に持つ場合は現在未解決であり，今後の課題とする．

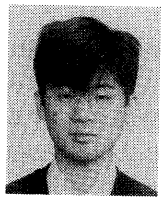
謝辞 本論文に対して有益な助言をいただいた査読者の方々に深く感謝いたします．

参考文献

1) Brown, G. M., Gouda, M. G. and Wu, C.: Token Systems that Self-Stabilize, *IEEE Trans. Comput.*, Vol. 38, No. 6, pp. 845-852 (1989).
2) Burns, J. and Pachl, J.: Uniform Self-Stabilizing Rings, *ACM Trans. Prog. Lang. Syst.*, Vol. 11, No. 2, pp. 330-344 (1989).
3) Dijkstra, E.: Self-Stabilizing Systems in Spite of Distributed Control, *Comm. ACM*, Vol. 17, No. 11, pp. 643-644 (1974).
4) Dijkstra, E.: Self-Stabilization in Spite of Distributed Control, *Reprinted in Selected Writing on Computing: A Personal Perspective*, pp. 41-46, Springer-Verlag, Berlin (1982).
5) Dolev, S., Israeli, A. and Moran, S.: Self Stabilization of Dynamic Systems Assuming Only Read/Write Atomicity, *Proceedings of the 9th ACM Symposium on Principles of Distributed Computing*, ACM (1990).
6) Ghosh, S.: Binary Self-Stabilization in Distributed Systems, *Inf. Process. Lett.*, Vol. 40, No. 3, pp. 153-159 (1991).
7) Herman, T.: Probablistic Self-Stabilization, *Inf. Process. Lett.*, Vol. 35, No. 2, pp. 63-67 (1990).
8) Huang, S.: Leader Election in Uniform Rings, *ACM Trans. Prog. Lang. Syst.*, Vol. 15, No. 3, pp. 563-573 (1993).
9) Israeli, A. and Jalfon, M.: Token Management Schemes and Random Walks Yield Self Stabilizing Mutual Exclusion, *Proceedings of the 9th ACM Symposium on Principles of Distributed Computing*, ACM (1990).
10) Kakugawa, H., Fujita, S., Yamashita, M. and Ae, T.: A Distributed k-Mutual Exclusion

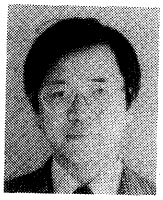
Algorithm Using k-Coterie, *Inf. Process. Lett.*, Vol. 49, No. 4, pp. 213-218 (1994).
11) Kakugawa, H., Fujita, S., Yamashita, M. and Ae, T.: Availability of k-Coterie, *IEEE Trans. Comput.*, Vol. 42, No. 5, pp. 553-558 (1993).
12) Kessels, J.: An Exercise in Proving Self-Stabilization with a Variant Function, *Inf. Process. Lett.*, Vol. 29, No. 1, pp. 39-42 (1988).
13) Lamport, L. and Lynch, N.: Distributed Computing: Models and Methods, van Leeuwen, J. (Ed.), *Handbook of Theoretical Computer Science* Vol. B, *Formal Methods and Semantics*, pp. 1157-1200, The MIT Press/Elsevier, Cambridge, Massachusetts/Amsterdam, Netherlands (1990).
14) Lin, C. and Simon, J.: Observing Self-Stabilization, *Proceedings of the 11th ACM Symposium on Principles of Distributed Computing*, ACM (1992).
15) 真鍋義文，青柳滋己：分散 k-相互排除問題について，信学技報，COMP 91-13 (May 1993).
16) 増澤利光，片山喜章：自己安定アルゴリズムについて，情報処理，Vol. 34, No. 11, pp. 1358-1365 (1993).
17) 西川直樹，増澤利光，都倉信樹：相互排除問題を解く均一な自己安定分散アルゴリズム，信学論，Vol. J75-D-I, No. 4, pp. 201-209 (1992).
18) Raymond, K.: A Distributed Algorithm for Multiple Entries to a Critical Section, *Inf. Process. Lett.*, Vol. 30, No. 4, pp. 189-193 (1989).
19) Raynal, M.: A Distributed Solution to the k-out of-M Resources Allocation Problem, *Lecture Notes in Computer Science*, p. 497, Springer-Verlag, Berlin (1991).
20) Schneider, M.: Self-Stabilization, *ACM Comput. Surv.*, Vol. 25, No. 1, pp. 45-67 (1993).
21) Srimani, P. K. and Reddy, R. L.: Another Distributed Algorithm for Multiple Entries to a Critical Section, *Inf. Process. Lett.*, Vol. 41, No. 1, pp. 51-57 (1992).

(平成5年8月25日受付)
(平成6年2月17日採録)



角川 裕次（正会員）

平成2年山口大学工学部電子工学科卒業．平成4年広島大学大学院工学研究科情報工学専攻博士課程前期修了．平成5年同博士課程後期中退．同年広島大学工学部第二類助手．分散アルゴリズム，分散処理に興味を持つ．



山下 雅史（正会員）

昭和49年京都大学工学部情報工学科卒業．昭和52年同大大学院修士課程修了．昭和55年名古屋大学大学院博士課程修了．工学博士．豊橋技術科学大学助手を経て，現在，広島大学工学部第二類教授．この間，昭和61年より約1年間，カナダサイモンフレーザー大学客員教授．マルチプロセッサの負荷分散問題，画像処理の基礎，組合せ問題の研究に従事．電子情報通信学会会員．