数理解析研究所講究録 第2088巻 2018年 14-22

個体群プロトコルにおける分割問題の一般化と空間複雑性について*

海野友希 北村直暉 泉泰介
名古屋工業大学
Tomoki Umino, Naoki Kitamura, Taisuke Izumi
Nagoya Institute of Technology

概 要

近年，受動的モバイルシステム，自然計算等の理論モデルとして，個体群プロトコルモデルが注目を集めている．本研究では，同一の状態から開始するエージェント群を与えられた整数構成比 $R = a_1 : a_2 : \cdots : a_k$ のサイズを持つ k 個のグループに分割する問題 (R-一般化分割問題) を提案し，その解法を与える．構成比がすべて均等な分割問題 (一様分割問題) に関しては既知のアルゴリズムが知られており，R-一般化分割問題は一様分割問題のプロトコルを用いて解くことが可能であるが，本研究では，そのようなアプローチを用いた場合に比べてエージェントの状態数が大幅に小さいプロトコルを提案する．提案プロトコルは，$a = \max_i a_i$ とすると，1 エージェントあたり $O(k \log a)$ の状態数で R-一般化分割問題を解く．

1 はじめに

1.1 研究の背景と本研究の成果

自然界において，同一種の個体の集合が成長，活動の過程で異なる機能を持つ個体群に分化していくことはしばしば観察される現象である．例えば蟻の群れからなるコロニーでは，規模の増大に伴って各個体が複数のグループへと自律的に分かれて役割分担を行うことが知られている．また，生物の幹細胞は，その分裂の仮定において機能分化が進んでいく．これらの現象において系全体がバランスを失わないために，個々の機能毎にその個体群の数（量）を適切にコントロールされることが不可欠である．機能分化の過程と共に個体群が自律的にそのようなコントロール機構を発現することは興味深い点であり，その背景にある根源的なメカニズムを解明することは上述のような現象の理解において重要であると考えられる．その一方で，近年，自然科学的な現象を計算機科学の文脈において捉えなおす試みが勃興しつつある．特に，多体系の生命・化学現象を分散アルゴリズムの観点から理解するアプローチは理論計算機科学と生命科学の境界領域として注目を集めている．そのようなアプローチにおいては，生命科学等における何らかの現象に対し，それを実現する分散アルゴリズムを設計し，その計算量的な複雑性を測ることで，観察対象の原理的な理解を深めることが企図されている．

これらの背景を動機として，本研究では，特に個体群プロトコル上における機能分化メカニズムの実現について検討する．個体群プロトコルモデル [1] は分散アルゴリズム理論におけるモバイル計算機網のモデルの一つであるが，近年，群知能，化学反応系のような，個々の個体が極めて限定

*本研究は JST 戦略的国際共同研究プログラム（SICORP）の支援を受けている．

された能力を持つようなシステムの理論モデルとして注目されている．このモデルでは，センサを取り付けられた移動体をエージェントとし，エージェントの集合を個体群とする．エージェントは他のエージェントが十分近くに接近した場合にのみ通信することができ，その通信によってお互いの状態を遷移させていく．個体群プロトコルモデルはセンサネットワークや分子ロボットなどの様々なシステムに適用することができる．

分散アルゴリズム理論的には，機能分化のメカニズムは，同一の初期状態から実行を開始するエージェント群を，獲得する機能ごとにグループ化する，一種の分割問題とみなすことができる．このとき，機能ごとに構成されるグループのサイズは一般には一様とは限らない．ただし，個々のエージェントは，システム中のエージェントの総数に関しての情報を持たないため，プロトコルによって構成されるグループのサイズを絶対的なエージェント数で指定することはできない．そのため，自然な問題設定として，本研究では，任意の整数構成比 $R = (a_1, a_2, \ldots, a_k)$ が与えられた時に個体群をその比に従って k 個のグループに分割する R-一般化分割問題を取り扱う．

本研究では個体群を構成するエージェント数が $\sum_{i=1}^{k} a_i$ に比例すると仮定した時に，先行研究のプロトコル [5] を応用したプロトコルよりも少ない状態数で分割するプロトコルを考案した．提案プロトコルで任意の整数構成比 $R = (a_1, a_2, \ldots, a_k)$ に分割するために必要な状態数は $O(k \log_2 \max_i a_i)$ となる．

1.2 既存研究

個体群の分割問題は，初めに一様 2 分割問題 [4] が安見らによって研究された．その後安見らは一様 2 分割問題の一般化である一様 k 分割問題 [5] についても研究を行い，個体群を比が一様な k 個のグループに分割するための状態数が $3k - 2$ のプロトコルを示した．一様 k 分割を行うプロトコルを用いて，任意の整数構成比に従ってグループを分割することは各エージェントを $\sum_i a_i$ 個のグループ一様分割したのち，状態を適切に（局所的に）デコードすることで可能であるが，このとき必要な状態数は $O(k \max_i a_i)$ の状態数が必要となる．

2 諸定義

2.1 個体群プロトコルモデル

個体群プロトコルモデルとは，受動的モバイルセンサネットワークの理論モデルの一つである．個体群プロトコルモデルでは個体群は N 個の有限状態機械で構成され，スケジューラによって選択された 2 つのエージェントが相互通信 (インタラクション) を行い，プロトコルに従って互いの状態を遷移させる．個体群プロトコルモデルの個体群は，個体群を形成するエージェント集合を V，通信可能なエージェント同士をつなぐ辺集合を E とした無向グラフ $G = (V, E)$ で表すことができる．この無向グラフ G を通信可能性グラフという．本研究では，以降 G は完全グラフであることを仮定して議論する．また，本稿ではエージェントを G の頂点集合 V の要素として, 通信を辺集合 E の要素として参照する．本研究では各エージェントは匿名であると仮定する．すなわち，各エージェントを V の要素として参照することはプロトコル説明の便宜上導入されているものであり，各エージェントは自身がどのような値によって参照されるかどうかを知ることはできない．形式的には，匿名性は，任意のエージェント $i (i \in V)$ について，その動作が i の値とは独立なプロトコルのみを考えるという仮定として定義される．

一般的な個体群プロトコルモデルは $P = (X, Y, Q, I, O, \delta)$ で表される．X はプロトコル開始時

に各ノードに与えられる入力文字の集合を表し，Y は各エージェントが出力する文字の集合を表す．Q はエージェントの状態集合を表す．各エージェントの初期状態は自身の入力文字にのみ依存して決定される．$I : X \to Q$ は入力アルファベットから状態集合への写像であり，各入力値に対応するエージェントの初期状態を定める．$O : Q \to Y$ は状態集合から出力アルファベットへの写像である．$\delta : Q \times Q \to Q \times Q$ は状態遷移関数を表し，インタラクトする 2 つのエージェントはこの関数に従って状態を遷移させる．任意の 4 つの状態 $q_1, q_2, q_3, q_4 \in Q$ に対して，$\delta(q_1, q_2) = (q_3, q_4)$ が成り立つとき，$(q_1, q_2) \to (q_3, q_4)$ を遷移と呼び，$\delta_1(q_1, q_2) = q_3, \delta_2(q_1, q_2) = q_4$ が定義される．本稿で検討する分割問題では，すべてのエージェントに同じ文字が入力文字として与えられ，初期状態は一意に定まることを前提としている．そのため，入力集合 X，写像 I の代用として，ある状態 $s \in Q$ を初期状態としてプロトコルを定義するものとする．すなわち，本稿ではプロトコル P は (Y, Q, s, O, δ) の 5 項組で定義するものとする．

ある時点での個体群を構成する各エージェントの状態への写像 $C : V \to Q$ をその時点での状況と呼ぶ．プロトコル P において状態 C は以下の条件を満たすとき，ある 2 つのノード u, v の通信 $e \in E$ によって C' へ遷移可能といい，$C \xrightarrow{P} C'$ と表す．

$$
\begin{aligned}
C'(u) &= \delta_1(C(u), C(v)) \\
C'(v) &= \delta_2(C(u), C(v)) \\
C'(w) &= C(w)(w \in V \setminus \{u, v\})
\end{aligned}
$$

文脈上プロトコル P が明確である場合，P を省略し $\to$ を用いる．また ，2 つの状況 C_0, C_m の間に遷移可能な状況列 $C_0 \to C_1 \to C_2 \to \cdots \to C_m$ が存在する時，C_m は C_0 から到達可能といい，$C_0 \to^* C_m$ と表す．状況 C におけるにおける出力アルファベットが $y \in Y$ であるようなエージェントの個数を $\#_y(C)$ で表すとし，任意の出力アルファベットの列 $Y' = y_1, y_2, \cdots, y_n$ に対して $H_{Y'}(C) = (\#_{y_1}(C), \#_{y_2}(C), \cdots, \#_{y_n}(C))$ とする．個体群プロトコルの状況 C における出力は $Out = H_Y(C)$ である．

2.2 実行の公平性

ある無限の状況の系列 $E = C_0, C_1, C_2, , , ,$ が，任意の $i (i \geq 0)$ について，$C_i \xrightarrow{P} C_{i+1}$ を満たす時，E を個体群プロトコル P の実行と呼ぶ．個体群プロトコルにおいては，各エージェント対がどのような順序でインタラクションを行うかはシステム自身がコントールできない．そのため，個体群プロトコルでは，一般に任意の実行においてシステムが求解状況へと到達し，安定することが求められる．一方で，真に任意の実行においてアルゴリズムが求解状況へと到達することは自明に不可能である．例えば，ある状況において，状況を変化させないようなインタラクションを行うエージェント対が存在すると，そのエージェント対が無限に通信を行うような実行ではアルゴリズムの実行が一切進まなくなる．そのため，可能な実行に関して何らかの制約を課すことが本質的に不可避である．本研究では，個体群プロトコルの設計において一般的に用いられる，大域公平性を導入する．

定義 1. 無限長の実行 E において，ある状況 C が無限にしばしば現れ，C から遷移可能なすべての状況が同じく無限にしばしば E に現れる時，この実行 E は大域公平性を満たすと呼ぶ．

大域公平性が保証していることは，直感的には，実行においてライブロックを生じないということである．例を挙げて説明する．あるプロトコル P をエージェント数 n のシステムで実行する

ことを考える．このとき可能な状況全ての集合を $\mathcal{C}_n$ とし，$\mathcal{T}_n = \{(C, C') | C \to C'\}$ としたとき，$\mathcal{G}_n = (\mathcal{C}_n, \mathcal{T}_n)$ で定義される有向グラフを n エージェントシステムにおける P の実行グラフと呼ぶ．このシステムにおける任意の実行は $\mathcal{G}_n$ 上の道に対応付けられるが，大域公平性は $\mathcal{G}_n$ 中の任意の閉路 H について，そこから外れる遷移が存在する限り，実行が H を無限に循環することがないことを保証している．次に，大域公平性の下での一般化分割問題を解くプロトコルの正当性について議論する上で有用な，以下の補題 [2,3] を導入する．

補題 1. 任意の n についてのプロトコル P の実行グラフ $\mathcal{G}_n = (\mathcal{C}_n, \mathcal{T}_n)$ について，ある状況の部分集合 $\mathcal{D} \subseteq \mathcal{C}_n$ をプロトコル P の正当な状況と定めるとする．もし G_n が以下の 2 条件を満たすならば，P は大域公平性のもとで必ず正当な状況に収束し，その後正当な状況を保ちつ続ける．

- 初期状況 C_0 から到達可能な任意の $C \in \mathcal{C}_n$ について，ある状況 $D \in \mathcal{D}$ が存在し，$C \to^* D$ が成立する．
- 任意の状況 $D \in \mathcal{D}$ について，$D \to^* C$ であるような任意の C について $C \in \mathcal{D}$

2.3　一般化分割問題

一般化分割問題とは，与えられた個体群を個体群プロトコルによって複数のグループに指定された構成比で分割する問題である．一般化分割問題は，構成比を指定する整数の組により定義される．$R = (a_1, a_2, \cdots, a_k)$ を構成比とする一般化分割問題を特に R-一般化分割問題と呼ぶ．この時，一般性を失わず $\gcd(a_1, a_2, \cdots, a_k) = 1$ と仮定する．一般化分割問題では，出力アルファベット $Y = \{g_1, g_2, g_3, \ldots, g_k\}$ をエージェントの分割先グループ名の集合とすると，状況 C において i 番目のグループに属するエージェントの数は $H_Y(C)_i$ で表すことができる．この Y に各エージェントの状態を出力関数 O によって写像する．この時，O は単射である必要はない．直観的には一般化分割問題は，構成比の総和を $\hat{a}$ としたとき，$H_y(C)_i = a_i N/\hat{a}$ が任意の i について所望の構成比を達成しているような状況に収束させるような問題とみなせるが，エージェント数 N が構成比の総和 ($\hat{a}$ とする) で割り切れるとは限らないため，一般にこの式の右辺は整数値とはならない．そのため，より正確には，全エージェントのうち，端数に相当する $N \mod \hat{a}$ 個のエージェントを適切に取り除いたとき，残りのエージェントが上記の式を満たしているとき，正しく分割された状況として定義する（すなわち，端数の振る舞いに関してはなにも制約を課さない)

定義 2. 与えられた R-一般化分割問題の出力アルファベット集合を Y とする．状況 C からある適切な $N \mod \hat{a}$ 個のエージェントを除外した状況 C' が存在し，任意の $i \in Y$ について

$$H_Y(C')_i = \frac{a_i}{\hat{a}} N \tag{1}$$

が成り立つとき，C は正当である．

なお，以降，$a = \max_{i \in [1,k]} \{a_1, a_2, \cdots, a_k\}$ と略記する．

3　提案プロトコル

本節では，R-一般化分割問題を状態数 $O(k \log_2 a)$ で解くプロトコルとその正当性について述べる．

3.1 提案プロトコルのアプローチ

本稿で提案するプロトコルは，各グループ i 毎に a_i 個のエージェントの集合体 (サブクラスタ) を作り，最後に各グループのサブクラスタをひとつにまとめた集合体 (メインクラスタ) を作ることを同時並行的に行うことで，所望の整数構成比を達成する．各エージェントの状態は，自身が所属するグループの ID，ならびに自身が所属している (グループ) クラスタ内に存在するエージェント数を計上するためのカウンタ，エージェントのステータスを表す変数の 3 項組から成る．エージェントステータスは自由状態，主状態，従属状態の 3 種類に分けることができる (実際には実装上の都合から主状態には複数の状態が割り当てられているので，ステータス変数は 3 値以上の値をとる．詳細は後述)．初期状態において，最初に任意のエージェントはグループ ID が 0，値が 0 の状態を持つ自由エージェントとして初期化されている．自由エージェントはある種の未分化状態であり，どのグループにもまだ属していないとみなす．また，自由エージェントにおいてはカウンタ値は特に意味を持たず，必ずゼロに初期化されていることが保証されている．2 つの自由エージェント同士がインタラクトすると，そのうちの一つはグループ 1，カウンタ値 1 の主エージェントに変化する．主エージェント i は，クラスタ構成におけるリーダの役割を果たし，他の主エージェント j とインタラクトしたときにそれを従属させる (すなわち，j のステータスを従属エージェントに変化させる) ことでサブクラスタのメンバに含めていく．このとき，主エージェント i のカウンタ値に従属されたエージェント j のカウンタ値を加算する．この処理は，j が過去に従属させたエージェント (と j 自身) を，間接的に i が従属させたとみなすことを意味する．このことより，任意の状況において，主エージェント i のカウンタ値は，i が (直接的あるいは間接的に) 従属させたエージェント数 $+1$ に一致している ($+1$ は自分自身を計上しているため)．また逆にすべての従属エージェントはある一つの主エージェント（自分を (直接的あるいは間接的に）従属させたエージェントに) 対応付けることができるため，グループ ID が 1，カウンタ値が k の主エージェントが存在することは，そのエージェントをある種のリーダとした，k エージェントのグループが構成されているとみなすことができる．このことより，最終的にカウンタ値 a_1 の主エージェント i が生成された時点で，グループ 1 のサブクラスタ構成が完了したことになる．このとき，エージェント i が新たな自由エージェント j とインタラクトしたならば，j はグループ 2 のカウンタ値 1 の主エージェントとなる．より一般的には，グループ x の主エージェント i は，グループ $x+1$ の主エージェントの生成者となる．以降，グループ $2, 3, \cdots$ と同様に処理していくことで，徐々にシステムはいくつかのサブクラスタに分割されていく．

今，各グループ内について，少なくとも 1 つ以上のサブクラスタが生成されているとする．このとき，グループ 1 のあるサブクラスタ内におけるリーダエージェント (カウンタ値 a_1 の主エージェント) は，グループ 2 以降のサブクラスタを順次ひとつづつ吸収合併してく．吸収されたサブクラスタのリーダエージェントのステータスを従属状態に変更することで，他のサブクラスタに同時に吸収されないことを保証する．最終的にグループ 2 からグループ k までのサブクラスタをひとつづつ吸収した後で，吸収後のグループ 1 のリーダは自身を従属状態として，メインクラスタの構成を完了する．このクラスタはすべてのエージェントが従属状態となるため，以降影響を受けることはない．

以上が，本研究における提案プロトコルの概要であるが，実際にプロトコルが正しく動作するためには，まだいくつか検討すべき事項が存在する．

1. 一つのサブクラスタの生成時に，中途半端に大きいカウンタ値を持つエージェントが多数生成されてデッドロックを生じる恐れがある．たとえば，サブクラスタのサイズが $a_1 = 5$ の時に，すべての主エージェントがカウンタ値 3 を持ってしまい，かつ自由エージェントの個数

がゼロであるとする．このとき，いかなる主エージェントをインタラクトさせてもサブクラスタサイズ 5 のサブクラスタを生成することができないため，デッドロックに陥る．

2. 一つ目のケースに類似するが，ある特定グループのサブクラスタが過剰に多く生成される可能性がある．例えばすべてのエージェントがグループ 1 のサブクラスタを生成してしまい，それ以外のグループのサブクラスタを一切生成しないという状態に陥る恐れがある．

3. 上述のメカニズムをナイーブに実装すると，グループ i のエージェントのカウンタ値として a_i 通りの値が必要となるため，必要な状態数が大きくなる．

以降，これらの問題を解決するためのアイデアを提示する．問題点 1，2 に関しては，本質的な問題は一度構成されたサブクラスタ（あるいはその途中で生成されるエージェントのグループ) を解体できない点にある．そのため，本提案プロトコルでは，サブクラスタを解体可能とするために，上述のクラスタ構成における状態遷移をすべて可逆化する．すなわち，ある遷移 $(a, b) \to (c, d)$ に対して，逆の遷移 $(c, d) \to (a, b)$ を追加する．ただし，プロトコルの安定性を保証するために，メインクラスタの構成を完了させる最後の遷移のみは不可逆とする（ただし，一部の遷移に関しては可逆性を保証するために追加した遷移が他の遷移と干渉するため，その場合については別のメカニズムで可逆化を実現する．詳細は後述)．こうすることで，プロトコル実行中に構成される任意のサブクラスタは解体可能となるため，デッドロックが生じることを避けることができる．可逆化することで，システムがサブクラスタの構成，解体を繰り返すようなループ実行が生じうるが，大域公平性を仮定している限りそのような無限ループの存在はプロトコルの収束性に影響を与えないことに注意する必要がある．

問題点 3 に関しては，グループ i が取りうるカウンタ値として，0 から a_i を許すのではなく，その一部の値のみカウンタ値として許すことで，状態数の削減を行う．具体的には，以下の式で与えられる集合 $A(a)$ を導入する: 任意の整数 a の 2 進数表現を $(a)_2$ とし，$(a)_2$ の i 番目の値を i 番目の要素とするベクトルを $\vec{a}$ とし，a のビット列の桁数を m とする．このとき，$A(a)$ を以下のように定義する．

$$A(a) = \{x \in \mathbb{N} \mid \exists j \in \mathbb{N} : x = 2^j \wedge x \leq a\} \cup \left\{ x \in \mathbb{N} \;\middle|\; \exists j \in \mathbb{N} : x = \sum_{h=1}^{j} \vec{a}_h \times 2^{h-1} \wedge x \leq a \right\}$$

直観的に説明すると，集合 $A(a)$ は，a を超えない 2 のべき乗で表現可能な値，および任意の $h \leq m$ について，$(a)_2$ の下位 h ビットをゼロでマスクした値，の 2 種類により構成されている．定義より $A(a)$ のサイズは明らかに $O(\log_2 a)$ で収まる．提案アルゴリズムでは，各グループ i について，カウンタ値として $A(a_i)$ 中の値のみを用いる，$A(a_i)$ 中のカウンタ値のみでサブクラスタが構成可能であることは以下のように主エージェントをインタラクトさせる過程が存在することから容易に確認できる．

- まず，任意の $\vec{a}_i = 1$ であるような，i について，カウンタ値 2^i の主エージェントを作成する．任意の x について，2^x のカウンタ値は，カウンタ値 2^{x-1} の主エージェント 2 つをインタラクトさせることで生成可能なので，この生成過程は 2 のべきのカウンタ値のみを用いて実現可能である．

- 次に，最初のステップで構成された主エージェント群をカウンタ値の大きい順に順次足し合わせていく，この過程において，h 回の足し合わせで生じるカウンタ値は $(a)_2$ において値の上位 h 個のビット 1 を残し，残りをゼロでマスクした値となるので，$A(a)$ 中のカウンタ値のみで実現可能である．

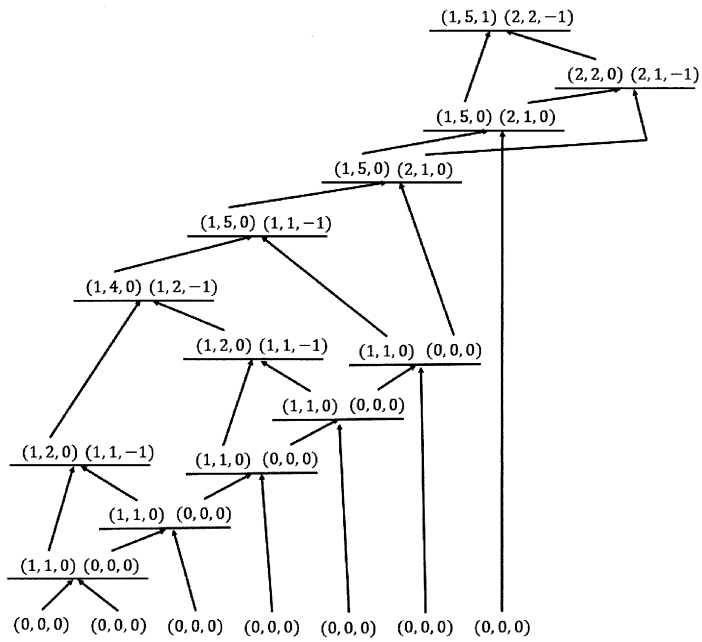

図 1: 5 : 2 分割における 1 つのクラスタ構成の例

上記プロセスの例として，5:2 分割におけるメインクラスタ構成の過程を図 3.1 に示す．

3.2 プロトコルの詳細

提案プロトコルでは，エージェントの状態を $(gid, v, state)$ の 3 項組で表す．gid はエージェントが分割されるグループの ID を表し，v はカウンタの値を表す．そして，$state$ はエージェントのステータスを表す．$state = -1$ は従属エージェントであることを表し，$state = 0$ かつ $v = 0$ は自由エージェントであることを表す．それ以外の場合はエージェントが主エージェントであることを表している．一般に $gid = 1$ の時を除いて，主エージェントの $state$ 値は 0 である．$gid = 1$ のエージェントに関しては，特に $v = a_1$ のとき，構成されたサブクラスタを順次吸収合併していくため，何番目のグループまで吸収したかを記憶しておく必要がある，$state$ はそのためのカウンタとしても働く．前述の通り，カウンタ値 v は任意の値を取ることはできない．実際に可能なグループ i のエージェント状態集合 $Q_a^{g_i}$ を以下のように定義する．

$$Q_a^{g_i} = \{(i, x, 0), (i, x, -1) \mid x \in A(a)\}$$

となる．ただし，$a = 1$ の場合は $Q_a^{g_i}$ に $(i, 2, 0)$ を追加する．以上のもと，R-一般化分割問題を状態数 $O(k \log_2 a)$ で解くプロトコル P_{div} を以下に示す．

ここからは，遷移関数の各規則の役割について述べる．1 行目はグループ 1 の主エージェントを生成する規則に相当する．他のグループの主エージェントの生成は 2,3 行目の規則によって開始される．主エージェント同士がインタラクトして，片方を従属させる処理は 4 行目の規則が相当す

Algorithm 1 P_{div}

Require: $1 \leq i \leq k$

Require: $0 \leq s \leq k-2$

$$Y = \{g_1, g_2, \ldots, g_k, T\}$$

$$Q = \bigcup_{i=1}^{k} Q_{a_i}^{g_i} \cup \{(1, a_1, x) \mid 1 \leq x \leq k-1\} \cup \{(0,0,0)\}$$

$$s = (0,0,0)$$

$$O = \{q \in Q_{a_1}^{g_1}\} \cup \{(1, a_1, k-1)\} \to g_1, \ldots, \{q \in Q_{a_k}^{g_k}\} \to g_k,$$
$$\{(0,0,0)\} \cup \{(1, a_1, x) \mid 1 \leq x \leq k-2\} \to T\}$$

δ の構成規則:

1: $(0,0,0)(0,0,0) \to (1,1,0)(0,0,0)$
2: $(i,x,0)(0,0,0) \to (i,x,0)((i \bmod k)+1,1,0)$
3: $(1,a_1,x)(0,0,0) \to (1,a_1,x)(2,1,0) \quad s.t. 1 \leq x \leq k-1$
4: $(i,x,0)(i,y,0) \to (i,x+y,0)(i,y,-1) \quad s.t.(i,x,0),(i,y,0),(i,x+y,0) \in Q_{a_i}^{g_i}$
5: $(i,2,0)(i,x,-1) \to (0,0,0)(0,0,0)$
6: $(i,x,0)(i,y,-1) \to (i,x_1,0)(i,x_2,0) \quad s.t. x = x_1 + x_2 \wedge (i,x_1,0),(i,x_2,0) \in Q_{a_i}^{g_i}$
7: $(1,a_1,s)(s+2,a_{s+2},0) \to (1,a_1,s+1)(s+2,a_{s+2},-1)$
8: $(1,a_1,s)(s+1,a_{s+1},-1) \to (1,a_1,s-1)(s+1,a_{s+1},0)$

る．この時カウンタの加算が生じるが，加算結果が可能なカウンタ値となる場合のみ，この規則は適用可能である．5,6 行目は上述のインタラクションの可逆化のために追加する規則である．注意する点として，5 行目の，グループに属するエージェントを自由エージェントへと戻す可逆な遷移は，元の遷移をそのまま反転させると他の遷移と干渉してしまうため，単純に可逆化できない．そのため，カウンタ値 2 のエージェントと一つの従属エージェントをまとめて 2 つの自由エージェントに戻すという処理で可逆化を実現している．このとき，カウンタ値 1 のエージェントが一つだけグループ内に残る状況が生じうるが，そのような状況はプロトコルのデッドロックを引き起こさないことが保証できる．7 行目，8 行目はグループ 1 のリーダがメインクラスタ構成のために他のサブクラスタを吸収合併するための遷移，およびその可逆化のための遷移である．

3.3 プロトコルの正当性

本節では，P_{div} の正当性について議論する．詳しい証明は紙幅の都合上省略するが P_{div} について以下の 3 つの補題が成り立つ．

補題 2. 任意の状況 C において $\#_{(i,x,0)}(C) = m$ のとき，少なくとも $m(x-1)$ 個の $gid = i, state = -1$ を持つエージェントが存在する．

補題 3. P_{div} の任意の状況 C において，$\#_{1,a_1,s}(C) = m$ のとき，$1 < i \leq s+1$ について $gid = i, v = a_i, state = -1$ のエージェントが少なくとも m 個存在する．

補題 4. P_{div} は，$m(m \geq \hat{a})$ 個の $(0,0,0)$ の状態を持つエージェントが存在するとき，通信するエージェント対の順序を選ぶことで $\frac{m}{\hat{a}}$ 個の $(1,a_1,k-1)$ を生成することができる．

補題 2 はグループ i のカウンタ値が x の主エージェントが存在するときに，その主エージェントによって従属されたグループ i に属する従属エージェントが必ず $x-1$ 個存在することを保証している．すなわち，この補題によってサブクラスタの構成の正当性が保証される．補題 3 はグループ 1 の $state = s$ の主エージェントが存在するときに，その主エージェントによって従属されたグループ $i(1 < i \leq s+1)$ のカウンタ値 a_i の主エージェントが必ず存在することを保証している．つまり，この補題によってメインクラスタの構成の正当性が保証される．補題 4 は初期状態のエージェントが $\hat{a}$ 個以上存在していれば，所望の構成比を達成したメインクラスタを構成できることを保証している．上記の 3 つの補題から定理 1 を示すことができる (証明は紙幅の都合上省略する)．

定理 1. P_{div} は $N = c\hat{a}(c \in \mathbb{N})$ のとき，R-一般化分割問題を状態数 $O(k \log_2 a)$ で解くことができる．

参考文献

[1] Angluin, D., Aspnes, J., Diamadi, Z., Fischer,M.J., Peralta, R. Computation in networks of passively mobile finite-state sensors. . In: Proceedings of the 23rd annual ACM symposium on Principles of Distributed Computing. pp. 290?299 (2004) .

[2] Fischer M., Jiang H. (2006) Self-stabilizing Leader Election in Networks of Finite-State Anonymous Agents. In: Proceedings of the International Conference on Principles of Distributed Systems (OPODIS 2006) Pages 395–409

[3] Mohamed G. Gouda. The Theory of Weak Stabilization. In: Proceedings of the 5th International Workshop on Self-Stabilizing Systems Pages 114-123

[4] Yasumi H., Ooshita F., Yamaguchi K., Inoue M., Constant-Space Population Protocols for Uniform Bipartition. In: Proceedings of 21st International Conference on Principles of Distributed Systems (OPODIS 2017) Pages 19:1–19:17

[5] Yasumi H., Kitamura N., Ooshita F., Izumi T., Inoue M., A Population Protocol for Uniform k-partition under Global Fairness. In: Proceedings of IPDPS workshops(IPDPS-ADPCM), 2018 (to appear).