

# モバイルロボット群の整列問題における 作業領域と実行時間

清水 貴弘<sup>†1,a)</sup> 山内 由紀子<sup>†2,b)</sup>

**概要：**本研究では、自律的に移動するロボット群の分散協調問題として、各ロボットが持つ固有のキーの順にロボット群を整列する**整列問題**を考える。特に、整列問題を解くための実行時間とロボットが移動する範囲である作業領域に着目する。各ロボットがすべてのロボットの位置とキーを観測できる場合に対して、作業領域と実行時間が異なる5つのアルゴリズムを提案する。さらに、作業領域のサイズに対する実行時間の下界を示す。

**キーワード：**分散アルゴリズム, モバイルロボット, 実行時間, 作業領域

## Working Space and Execution Time of the Sorting Problem for Mobile Robots

**Abstract:** In this study, we consider a distributed coordination problem for a group of autonomous mobile robots, called the *sorting problem* in which the robots are required to form a line in the order of their unique keys. We focus on the execution time required to solve the sorting problem and the workspace in which the robots move. Assuming that each robot can observe the positions and keys of all robots, we propose five algorithms with different execution times and workspace requirements. Furthermore, we establish a lower bound on the execution time concerning the workspace size.

**Keywords:** Parallel and Distributed Algorithms, Distributed Algorithms, Robot Systems

### 1. はじめに

分散アルゴリズム理論では、自律的に移動するモバイルロボット群による**パターン形成**

**問題** [1], [2] や**集合問題** [3], [4] が研究されてきた。既存研究の多くでは、匿名 (識別不能) で共通の分散アルゴリズムを実行しながら自律的に移動するロボット群を想定している。さらに、各ロボットはメモリや共通の座標系、通信手段を持たないため、他のロボットの位置を観測することで協調動作を実現できるかどうか議論されてきた。

ロボットは、**観測-計算-移動サイクル**を

<sup>†1</sup> 現在, 九州大学工学部  
Presently with School of Engineering, Kyushu University

<sup>†2</sup> 現在, 九州大学大学院システム情報科学研究所  
Presently with Faculty of Information Science and Electrical Engineering, Kyushu University

a) shimizu.takahiro.602@s.kyushu-u.ac.jp

b) yamauchi@inf.kyushu-u.ac.jp

繰り返し実行することで動作する。各時刻  $t = 0, 1, 2, \dots$  に全てのロボットが同期してサイクルを実行する **完全同期モデル**を想定する。観測フェーズでは各ロボットは自身の局所座標系で他のロボットの位置とキーを観測する。計算フェーズでは、観測フェーズで得られた観測結果をもとに、各ロボットが自身の目的地を計算する。このように、観測結果のみから計算を行うロボットを**無記憶なロボット**という。移動フェーズでは、各ロボットは計算した目的地に向けて移動する。本研究では、全てのロボットが各時刻にサイクルの各フェーズを同期して実行する**同期モデル**を想定する。

本研究の目的は、一般的なアルゴリズム評価指標である**実行時間**と移動により生じる**作業領域**という観点から、モバイルロボット群の分散協調手法の新しい評価基準を提案することである。作業領域とは、任意の初期配置から始まるアルゴリズムの実行において、ロボット群が移動する領域のうち、最もサイズが大きいもののことである。既存の多くのパターン形成アルゴリズムや集合問題アルゴリズムでは、ロボット群は初期配置の最小包含円の外に出ないため、作業領域は着目されてこなかった [1], [2], [3]。また、可解性についての議論が多く、実行時間についての議論はあまりされてこなかった [4]。本研究では、ロボット群による**ソート問題**を新たに提案する。ソート問題とは、固有のキーを持つロボットの集合をそのキーの順番に整列する問題である。本稿では、ソート問題を解くための実行時間と実行中にロボットが移動する範囲である作業領域の関係を示す。

本研究では、ロボットが自身のキーと他のすべてのロボットの位置とキーを観測できる場合を考える。作業領域の異なる5つの提案アルゴリズムの実行時間を調べることで、作業領域のサイズが大きいほど実行時間が小さくなることを示す。さらに、 $n$  台のロボット群のソートにおける作業領域のサイズが  $h \times n$  ( $2 \leq h \leq n$ ) のとき、ロボット群のソートにおける実行時間の下界が  $\log_h n$  であることを示す。

## 2. 準備

整数値の**キー**をもつ  $n$  台のロボットを考える。同じ値のキーを持つロボットはいないとする。ただし、キーが1から  $n$  であるとは限らない。キーが  $i$  番目に小さいロボットを  $r_i$  とする。ロボットの集合を  $R = \{r_1, r_2, \dots, r_n\}$  とする。

各ロボットは直径1の円盤であり、近づけば衝突する。ロボット  $r_i$  の円盤の中心の座標を、 $r_i$  が存在する位置とする。ロボットの視界は無限であり、他のロボットの位置とキーを観測できるとする。また、ロボットは他のロボットの視界を妨げない。時刻  $t$  のときの大域座標系におけるロボットの位置とキーの集合  $C_t = \{(x_i, y_i, k_i) \mid r_i \in R\}$  を**ロボットの配置**という。

各ロボット  $r_i$  は大域座標系  $Z$  を知らず、自身の**局所座標系**  $Z_i$  で他のロボットの位置を観測する。 $r_i$  の局所座標系は右手系の直交座標系とし、その原点は  $r_i$  の現在の位置であり、 $x$  座標の方向と向きづけは任意であるとする。単位長はロボットの直径とする。

本研究では、各時刻  $t = 0, 1, 2, \dots$  に全てのロボットが同期して**観測-計算-移動サイクル**を実行する**完全同期モデル**を想定する。観測フェーズでは各ロボットは自身の局所座標系で他のロボットの位置とキーを観測する。計算フェーズでは、観測フェーズで得られた観測結果をもとに、各ロボットが自身の目的地を計算する。目的地を計算するアルゴリズムは全てのロボットで同一であり、 $A$  とする。移動フェーズでは、各ロボットは計算した目的地に向けて移動する。ロボットは途中で止まらず必ず目的地に到達する。ロボット  $r_i$  の目的地までの経路に他のロボット  $r_j$  がいる場合でも、 $r_j$  が同じ時刻に移動して  $r_i$  の経路からいなくなる時には衝突しない。

ロボット群がキーの昇順もしくは降順に一直線上に並んだ配置を**ソート問題の終端配置**と呼ぶ。時刻  $t$  の配置を  $C_t$  と表し、初期配置  $C_0$  から始まる配置の系列  $C_0, C_1, \dots$  をアルゴリズム  $A$  の**実行**と呼ぶ。ロボット群が

ソート問題の終端配置に到達したとき、 $A$ の実行が終了したという。任意の初期配置  $C_0$  から開始するアルゴリズム  $A$  の実行  $C_0, C_1, \dots$  が有限時間内にソート問題の終端配置に到達する時、この実行は**ソート問題を解く**と言う。与えられた初期配置  $C_0$  から始まるアルゴリズム  $A$  の任意の実行がソート問題を解く時、 $A$  は初期配置  $C_0$  からソート問題を解くと言う。

アルゴリズム  $A$  の **実行時間** を任意の初期配置から始まる実行のうち、最も長い実行の長さとする。ただし、ソート問題が解けない場合には、実行時間を無限大とする。アルゴリズム  $A$  の **作業領域** とは、任意の初期配置から始まる実行において、ロボット群が移動する領域のうち、最もサイズが大きいものこのことで、作業領域のサイズを作業領域の面積とする。提案アルゴリズムでは、作業領域が長方形になるので高さ  $h$  で横幅  $w$  の領域面積、 $h \times w$  とする。

本研究では、ロボット群が任意の順番で一直線上に並んだ初期配置のみを想定する。

### 3. ロボットのソートアルゴリズム

本章では作業領域と実行時間の関係を探るため、作業領域が異なる5つのアルゴリズムを提案する。

提案手法では、すべてのロボットが一直線上に隙間なく並んだ初期配置のみを考える。ここで、仮想的な座標系  $Z^*$  を考える。初期配置において  $r_1$  の位置から  $r_n$  の位置に向かう直線を  $Z^*$  の  $y = 0$ 、つまり  $x$  軸と考え、 $r_1$  から  $r_n$  へ向かう方向を  $x$  軸の正の向きとする。また、座標系の単位距離をロボットの直径である1とする。 $Z^*$  の  $x$  座標は初期配置において、ロボットがいる位置を  $x$  座標が小さい方から  $1, 2, \dots, n$  とする。各提案アルゴリズムでは、初期配置において定義した  $Z^*$  の原点と軸の向きが不変となるようになっている。

ロボットのキーを  $Z^*$  の  $x$  座標が小さいものから見たとき、キーが昇順にソートされているとき、ロボット群は**水平にソートされている**という。同様に、 $y$  座標に関しては、**垂直にソートされている**という。

表1に本章の結果を示す。提案アルゴリズムでは、作業領域のサイズが大きいほど実行時間が小さくなるという関係が見てとれる。特に3.6節では作業領域が  $h \times n$  のときの実行時間の下界を示す。

### 3.1 作業領域 $(n - 1) \times n$ の対角線ソートアルゴリズム

このアルゴリズムでは、まずキーが最小のロボット  $r_1$  と最大のロボット  $r_n$  を除いたすべてのロボットを垂直にソートし、 $r_2$  から  $r_{n-1}$  の水平方向に他のロボットがない配置にする。その後、ロボット  $r_i$  は  $Z^*$  の  $x$  座標が  $i$  となるように水平方向に移動し、水平にソートされた配置に到達する。その後、 $r_2$  から  $r_{n-1}$  が  $Z^*$  の  $x$  軸上まで移動して一直線に並ぶことでソートが完了する。

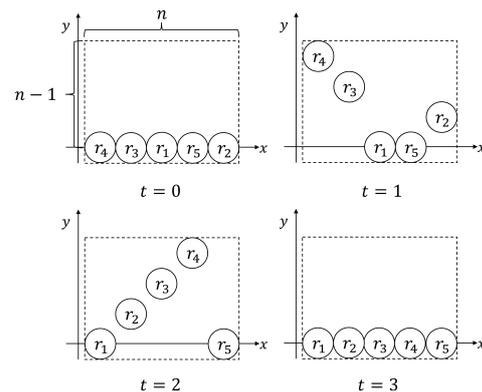


図1  $n = 5$  のときの対角線ソートアルゴリズムの実行例

Fig. 1 Execution Example of the Diagonal Sort Algorithm for  $n = 5$

図1に  $n = 5$  のときの実行例を示す。 $t = 0$  が初期配置であり、1ステップで時刻が1進む。この図では、横が  $n = 5$ 、縦が  $n - 1 = 4$  で作業領域のサイズは20である。時刻  $t = 3$ 、つまり3ステップでソートが完了していることがわかる。

**定理 3.1** 対角線ソートアルゴリズムはすべてのロボットが一直線上に並んだ初期配置からソート問題を解く。作業領域は  $(n - 1) \times n$  であり、完全同期モデルでの実行時間は3である。

表 1 提案アルゴリズムの作業領域と実行時間

提案アルゴリズム	作業領域のサイズ	実行時間
選択ソートアルゴリズム (3.2 節)	$2 \times n$	$2(n-2)$
交換ソートアルゴリズム (3.3 節)	$3 \times n$	$(3/2)n$
ハーフソートアルゴリズム (3.4 節)	$(n/2 + 1) \times n$	6
対角線ソートアルゴリズム (3.1 節)	$(n-1) \times n$	3
マージソートアルゴリズム (3.5 節)	$h \times n (h \geq 2)$	$2\lceil \log_h n \rceil + 1$

### 3.2 作業領域 $2 \times n$ の選択ソートアルゴリズム

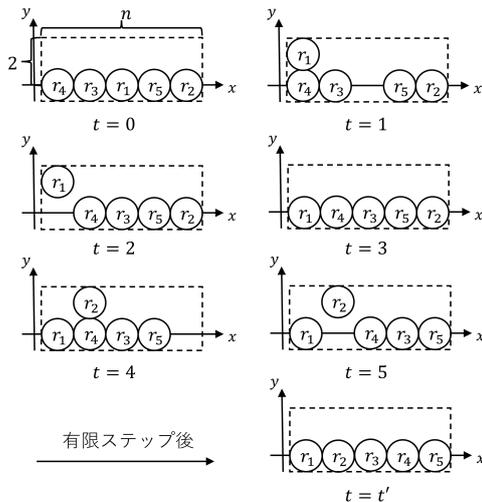


図 2  $n = 5$  のときの選択ソートアルゴリズムの実行例

Fig. 2 Execution Example of the Selection Sort Algorithm for  $n = 5$

このアルゴリズムでは選択ソートのように、キーが最小のロボット  $r_1$  から  $Z^*$  の  $x$  座標の小さい方に移動し順番にソートしていく。ロボット  $r_i$  が座標  $i$  に向けて移動を開始すると、元々ロボット  $r_i$  のいた場所には空間ができる。この空間を埋めるために、すでにソートされたロボット群を除く  $r_i$  より  $x$  座標が小さい位置にいたロボット群は一斉に  $Z^*$  の  $x$  軸正の向きに移動する。  $r_i$  は他のロボット群が一斉に  $Z^*$  の  $x$  軸正の向きに移動することで  $(i, 0)$  からロボットがいなくなるので、  $(i, 0)$  に移動することができる。  $r_i$  はソートされたロボット群の中で  $y$  座標が最も大きいロボットの隣に移動する。このとき、ソートされたロボット群の中には  $r_i$  がないので、  $r_i$  は  $(i, 0)$  に移

動するとき、現在地より右に行くことはない。

図 2 に  $n = 5$  のときのアルゴリズムの実行例を示す。  $t = 0$  が初期配置であり、1 ステップで時刻が 1 進む。この図では、横が  $n = 5$ 、縦が 2 で作業領域のサイズは 10 である。時刻  $t = 3$  で  $r_1$  が一番左に移動していることがわかる。また、3 ステップごとにロボットが一台ソートされていることがわかる。

**定理 3.2** 選択ソートアルゴリズムはすべてのロボットが一直線上に並んだ初期配置からソート問題を解く。作業領域は  $2 \times n$  であり、完全同期モデルでの実行時間は  $2(n-2)$  である。

### 3.3 作業領域 $3 \times n$ の交換ソートアルゴリズム

このアルゴリズムでは、ロボット  $r_i$  を  $y$  軸方向に 1 移動させ、  $Z^*$  における  $x$  座標が  $i$  になるまで移動させた後、  $Z^*$  の  $(i, 0)$  にいるロボット  $r_j$  を  $r_i$  のいない方に  $y$  軸方向に移動させて  $Z^*$  における  $x$  座標が  $j$  になるまで移動させるという操作を繰り返すことでソートを行う。  $r_i$  が  $Z^*$  の  $(i, 0)$  に移動するときロボット  $r_j$  も同じ時刻に移動するため、  $r_i$  と  $r_j$  は衝突しない。このように初期配置に対して、  $y$  座標が  $-1$  から  $1$  までの上下の空間を使うことで、選択ソートアルゴリズムの実行時間の短縮を図る。

図 3 に  $n = 5$  のときのアルゴリズムの実行例を示す。この図では、横が  $n = 5$ 、縦が 3 で作業領域のサイズは 15 である。時刻  $t = 1$  で  $r_1$  が一番左に移動し、  $t = 2$  で下に移動してソートされていることがわかる。また、  $t = 1$  からは時刻が 1 進むごとにロボットが一台ソートされていることがわかる。

**定理 3.3** 交換ソートアルゴリズムはすべ

でのロボットが一直線上に並んだ初期配置からソート問題を解く。作業領域は  $3 \times n$  であり、完全同期モデルでの実行時間は  $(3/2)n$  である。

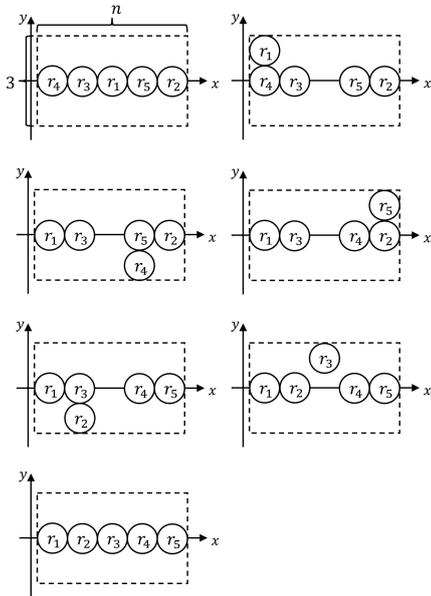


図 3  $n = 5$  のときの交換ソートアルゴリズムの実行例  
 Fig. 3 Execution Example of the Exchange Sort Algorithm for  $n = 5$

### 3.4 作業領域 $(\frac{n}{2} + 1) \times n$ のハーフソートアルゴリズム

このアルゴリズムでは、まずロボット群のキーの中央値よりもキーが大きいロボット群を  $x$  軸の正の向きに移動させて、対角線ソートアルゴリズムに似た手順でソートする。その後、中央値以下のロボット群を  $x$  軸の負の向きに移動させる。

図 4 に  $n = 5$  のときの実行例を示す。この図では、横が  $n = 5$ 、縦が  $5/2 + 1 = 3.5$  で作業領域のサイズが 17.5 である。時刻  $t = 0$  から  $t = 3$  までで中央値よりも大きいロボット群  $r_4, r_5$  が右側に移動してソートされていることがわかる。また、 $t = 3$  からは  $r_3$  から  $r_4$  までの 4 台で対角線ソートアルゴリズムを実行する。3 ステップごとに列が半分ずつソートされているのがわかる。

**定理 3.4** ハーフソートアルゴリズムはすべてのロボットが一直線上に並んだ初期配置からソート問題を解く。作業領域は  $(\frac{n}{2} + 1) \times n$  であり、完全同期モデルでの実行時間は 6 である。

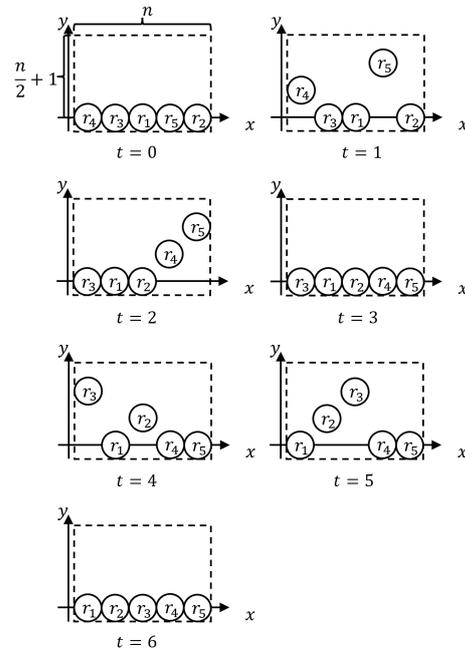


図 4  $n = 5$  のときのハーフソートアルゴリズムの実行例  
 Fig. 4 Execution Example of the Half Sort Algorithm for  $n = 5$

### 3.5 作業領域 $h \times n$ のマージソートアルゴリズム

このアルゴリズムではマージソートのように  $h^i$  ( $i = 1, 2, \dots$ ) 台ずつの組を作りソートしていく。まず、 $h$  台の組の中で垂直にソートする。次に、組の中で水平にソートする。このとき、ロボットの配置が 1 台ずつの階段状になる。その後、配置が 3 台ずつの階段状になるように  $y$  軸方向に移動する。次に、今度は  $h^2$  台を 1 組として、組の中で水平にソートする。これを全体がソートされるまで繰り返す。マージソートアルゴリズムでは、図 5 のように  $h^i$  個ずつの組が下から上に階段状に配置されている状態を第  $i$  フェーズ終了状態という。

図 6 に  $n = 5$  のときの実行例を示す. この

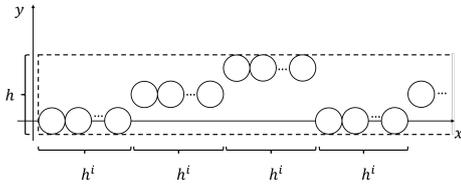


図 5 第  $i$  フェーズ終了状態  
 Fig. 5 End State of Phase  $i$

図では, 横が  $n = 9$ , 縦が  $h = 3$  で作業領域のサイズが 27 である. 時刻  $t = 1$  でそれぞれの組の中で垂直にソートされている.  $t = 2$  ではそれぞれの組の中で水平にソートして, 第 0 フェーズ終了状態になっている.  $t = 3$  では,  $y$  軸方向に移動して第 1 フェーズ終了状態になっている. そこからすべてのロボットが水平にソートされ,  $x$  軸上に移動することでソートが完了する.

ロボットの配置の集合  $C_{i,1}^5, C_{i,2}^5$  の添え字  $i$  が増えていくとソートが進んでいく.  $C_{\lceil \log_h n \rceil}^5$  に含まれる配置に到達したときソートが完了する.

以下はロボットの配置の定義である.

- $C_{0,1}^5$ : 左から  $h$  台の組の中で水平にソートされておらず, 一直線上に並んでいる
- $C_{0,2}^5$ : 左から  $h$  台の組の中で水平にソートされておらず, 組の中で同じ  $y$  座標にロボットが複数台いない
- $C_{i,1}^5$  ( $1 \leq i \leq \lceil \log_h n \rceil$ ): 第  $(i-1)$  フェーズ終了状態
- $C_{i,2}^5$  ( $1 \leq i < \lceil \log_h n \rceil$ ): 左から  $h^i$  台の組の中で水平にソートされている, かつ, 左から  $h^{i+1}$  台の組の中で水平にソートされていない, かつ, 第  $j$  ( $j \leq i$ ) フェーズ終了状態ではない
- $C_{\lceil \log_h n \rceil, 2}^5$ : 全体が水平にソートされている, かつロボット群が  $x$  軸上に一直線に並んでいない
- $C_{\lceil \log_h n \rceil}^5$ : 全体が水平にソートされている, かつすべてのロボットが  $x$  軸上にいる

補題 3.1 任意の  $i, j, k$  ( $i \neq j, k = 1, 2, l = 1, 2$ ) について,  $C_{i,k}^5 \cap C_{j,l}^5 = \emptyset, C_{i,k}^5 \cap C_{\lceil \log_h n \rceil}^5 = \emptyset$  である.

各ロボットは無記憶であっても, 現在の観測結果  $C_t$  から  $Z^*, C_t \in C_i^1$  となる  $i$  が一意に判別することができる. そして, Algorithm 1 が期待する動作をして, 必ず実行が進む.

**Algorithm 1** ロボット  $r_i$  におけるマージソートアルゴリズム

```

1: //  $C$ : 現在のサイクルでの  $r_i$  の観測結果
2: //  $(x, y)$ :  $Z^*$  における現在の  $r_i$  の座標
3: //  $p$ :  $r_i$  は左から  $p$  番目の組に属する
4: //  $j$ : 自分の属する組の中でキーが  $j$  番目
5:
6: if  $C \in C_{0,1}^5$  then
7:   // 全体を  $h$  台ずつの組に分ける
8:   return  $(x, y + (j - 1))$ 
9: end if
10:
11: if  $C \in C_{0,2}^5$  then
12:   return  $(x, p \bmod h)$ 
13: end if
14:
15: if  $C \in C_{k,1}^5$  then
16:   if  $k = 1$  then
17:     // 全体を  $h^k$  台ずつの組に分ける
18:     return  $(x, p \bmod h)$ 
19:   else if  $k \geq \log_h n$  then
20:     // 全体を  $n$  台の組とする
21:     return  $(j, y)$ 
22:   else
23:     // 全体を  $h^k$  台ずつの組に分ける
24:     return  $((p - 1)h^k + j, y)$ 
25:   end if
26: end if
27:
28: if  $C \in C_{k,2}^5$  then
29:   if  $k = \lceil \log_h n \rceil$  then
30:     return  $(x, 0)$ 
31:   else
32:     // 全体を  $h^k$  台ずつの組に分ける
33:     return  $(x, p \bmod h)$ 
34:   end if
35: end if
    
```

定理 3.5 マージソートアルゴリズムはすべてのロボットが一直線上に並んだ初期配置からソート問題を解く. 作業領域は  $h \times n$  であり, 完全同期モデルでの実行時間は  $2\lceil \log_h n \rceil + 1$  である.

**3.6** ロボット群のソート問題における実行時間の下界

ロボットの台数を  $n$  台, ロボットの作業領

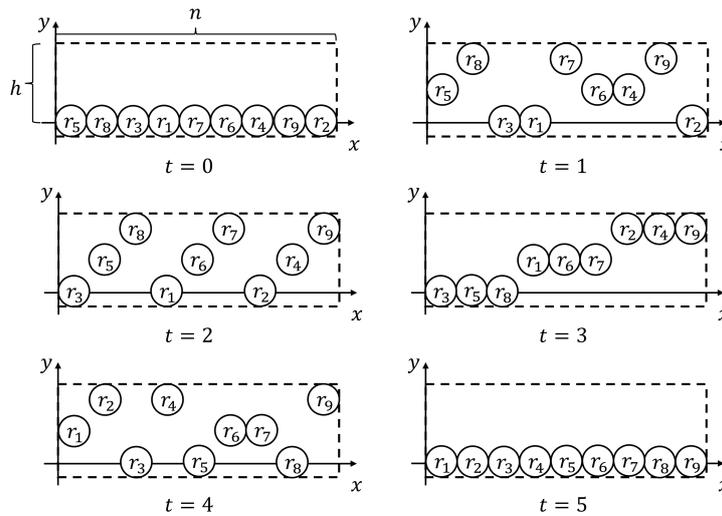


図 6  $n = 5$  のときのマージソートアルゴリズムの実行例  
 Fig. 6 Execution Example of the Merge Sort Algorithm for  $n = 5$

域を  $h \times n$  とする. 本節では, 逐次アルゴリズムと同様に決定木を用いてソート問題の実行時間の下界を調べる. ここで, 決定木はあるソートアルゴリズムが実行する比較を表現する木である. ロボットの直径が 1 なので, 図 7 のように同時に縦に並べられるのは高々  $h$  台である. ここで, 黒いロボットが図の右側の

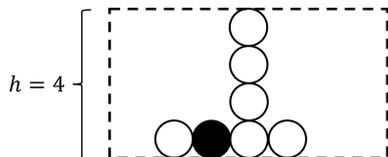


図 7 高さが  $h = 4$  の場合, 入れ替えられる最大数

空間に移動するには, 図で  $y$  軸方向に縦に並んでいる 4 台のロボット群の中で少なくとも 1 台が左右のどちらかによけて, 黒いロボットが移動できる空間を作る必要がある. よって, 作業領域の高さが  $h$  のとき水平方向に同時にソートできるロボット群の台数は高々  $h$  台である. また, それぞれの関係ごとに  $n/h$  個が同時に  $h$  値比較できる分散アルゴリズムを実行することを考えるので, 決定木の各内部節点は  $\frac{n}{h}$  個の  $h$  項組をラベルとして持つ.

このとき, それぞれの組の中のロボット群の並び方が  $h!$  通りあるので, 決定木は  $(h!)^{\frac{n}{h}}$  分木となる. また, 各葉は最終的な配置をラ

ベルとして持つ. 一直線に並んだ任意の初期配置からロボット群のソート問題を正しく解くソートアルゴリズムはサイズ  $n$  の入力に対して,  $n!$  個の葉が必要となる. 決定木の根から任意の葉までの道の長さの最大値, つまり決定木の高さ  $k$  がソートアルゴリズムの最大比較回数となる.

**定理 3.6** 作業領域が  $h \times n$  のとき,  $n$  台のロボット群の比較ソートアルゴリズムは最悪時に  $\Omega(\log_h n)$  回の比較が必要である.

#### 4. まとめ

本研究では, 各ロボットがすべてのロボットの位置とキーを観測できる場合に対して, 作業領域の異なる 5 つの提案アルゴリズムの実行時間を調べることで, 作業領域のサイズが大きいかほど実行時間が小さくなることを示した. 次に, 逐次アルゴリズムにおける比較ソートの比較回数の下界と同様の手法を用いて,  $n$  台のロボット群のソート問題を解くための実行時間の下界は, 作業領域のサイズを  $h \times n$  ( $2 \leq h \leq n$ ) としたとき,  $\Omega(\log_h n)$  となることを示した.

今後の課題は, より一般的な仮定の下でのソート問題の解法である. 本研究では完全同期モデルのみの議論にとどまっているため,

観測-計算-移動サイクルの実行タイミングに一切の仮定をおかない非同期モデル、さらに、完全同期モデルと非同期モデルの中間に位置する半同期モデルにおけるソート問題については未解決である。実際のロボットシステムではロボット間の同期が保証されない場合が多いため、非同期モデルに対応したアルゴリズムの開発が求められる。さらに、本研究では初期配置でロボット群が一直線に並んでいると仮定しているので、任意の初期配置からソートを行う分散ソートアルゴリズムの検討も必要である。

### 参考文献

- [1] Ichiro Suzuki and Masafumi Yamashita, “Distributed Anonymous Mobile Robots: Formation of Geometric Patterns,” *SIAM Journal on Computing*, Vol. 28, Issue 4, pp. 1347–1363, 1999.
- [2] Shantanu Das, Paola Flocchini, Nicola Santoro, and Masafumi Yamashita, “Forming sequences of geometric patterns with oblivious mobile robots,” *Distributed Computing*, Vol. 28, pp. 131–145, 2015.
- [3] Mark Cieliebak, Paola Flocchini, Giuseppe Prencipe, and Nicola Santoro, “Distributed Computing by Mobile Robots: Gathering,” *SIAM Journal on Computing*, Vol. 44, Issue 4, pp. 829–879, 2012.
- [4] Pavan Poudel and Gokarna Sharma, “Time-Optimal Gathering under Limited Visibility with One-Axis Agreement,” *Information*, Vol. 12, No. 11, pp. 448, 2021.
- [5] T. コルメン, C. ライザーソン, R. リベスト, C. シュタイン (共著), 浅野哲夫, 岩野和生, 梅尾博司, 山下雅史, 和田幸一 (共訳), 『アルゴリズムイントロダクション』, 第3版, 近代科学社, 2012.