

# エネルギー制限ロボットにおけるFCOMによるLUMIの効率的なシミュレーション

小原直将<sup>1</sup>    和田幸一<sup>2</sup>

<sup>1</sup> 法政大学大学院理工学研究科応用情報工学専攻

<sup>2</sup> 法政大学理工学部応用情報工学科

## 概要

本研究では、エネルギーに制限のあるロボットの動作をモデル化したスケジューラ（RSYNCH）上で自律的に動作するロボット群について考察する。他のロボットから観察できるが、自身では観察できないライトを持ったロボット群（FCOM）による、自身からも他のロボットからも観察できるライトを持ったロボット群（LUMI）のシミュレーションについて、既存のものより少ない色数でシミュレーション可能なアルゴリズムを設計した。既存のシミュレーションアルゴリズムでは、シミュレートされるアルゴリズムで使用するライトの色数が $L$ 色の時、 $256L * 16^L$ 色を用いるアルゴリズムでRSYNCHスケジューラ上でのFCOMによるLUMIのシミュレーションができるが、本研究で提案するシミュレーションアルゴリズムでは使用するライトの色数が $60L * 2^L$ 色にまで削減される。

## 1 はじめに

自律分散ロボットとは、自律的に動作する複数のロボットからなるシステムのことであり、適切なアルゴリズムにより、ロボット同士を協調して動作させ、与えられた問題を解く動作を実行させることができる [1]。その計算能力の分析の一環として、本稿ではRSYNCHと呼ばれる特殊なスケジューラ上で、2台以上のFCOMロボットで同数のLUMIロボットの動作をシミュレーションできるようなアルゴリズムについて検討した。先行研究により、このようなシミュレーションが可能であることは分かっており [2]、本研究ではライトの状態数を削減したアルゴリズムの設計とその妥当性の検証を行った。

## 2 ロボットの理論モデル

本稿におけるロボットは、二次元ユークリッド平面上を動く点として考える。ロボットはそれぞれが自律的に動作し、ロボットの集合全体に命令を出す中央制御装置は存在しない。各ロボットはそれぞれ独自のローカル座標系に従って動作する。ロボットは互いを区別するIDを持たず、外見で区別することもできず、全てのロボットが同じアルゴリズムを実行する。特に能力を追加されない限り、ロボットは他の

ロボットとの通信手段や過去の状態を記憶するメモリを持たない。ロボットは初期状態ではinactiveであり、事前に定義したスケジューラによって決められた時点でactivateされる。inactiveのとき、ロボットはアイドル状態となり休止している。各同期ラウンドでロボットがactiveになると、以下に示す3つの動作を1つのサイクル（LCMサイクル）として実行する。

1. Look: 自分の座標系を基準にして、全てのロボットの情報をスナップショットとして取得する。この動作は一瞬で行われ、ロボットは他のすべてのロボットを観測できると仮定する。
2. Compute: Lookで得られたデータを入力として全てのロボットに共通のアルゴリズムを実行し、出力として目的地を算出する。ロボットがライトを持っている場合は、ここで自身のライトの色を変更することができる。
3. Move: Computeで計算した目的地に向かって移動する。

これらのサイクルを繰り返すことで、ロボットは集団でタスクを実行し、与えられた問題を解決する。

以降では、与えられた問題を解決するためのアルゴリズムをアルゴリズム A と呼ぶ。観察可能な全てのライトは Look 処理で観察することができ、Compute 処理で利用、色の変更ができることとする。

### 3 ロボットのライトモデル

ロボットが持つライトの能力によって、以下のようなモデルが定義されている [1].

- LUMI(full-light) : 自身と他のロボットの両方から観察可能なライトを持つモデル.
- FSTA(internal-light) : 自身は観察可能だが、他のロボットからは観察できないライトを持つモデル.
- FCOM(external-light) : 自身は観察できないが、他のロボットからは観察可能なライトを持つモデル.

また、本稿で検討するモデルでは、各ロボットは時計回りの方向について合意していると仮定する。自身から見て時計回り方向で直前の位置にいるロボットを pred、直後の位置にいるロボットを suc と呼ぶ。

### 4 ロボットのスケジューラ

ロボットの動作を同期させる程度により、以下の3つのモデルが定義されている。

- FSYNCH(Fully Synchronous) : 全てのロボットは各ラウンドで完全に同期して LCM サイクルを実行する.
- SSYNCH(Semi Synchronous) : 各ラウンドで動作しないロボットが存在しうるが、動作するロボットは完全に同期して LCM サイクルを実行する.
- ASYNCH(Asynchronous) : 各ロボットは、同期せずに Look, Compute, Move 動作を繰り返す.

また、先行研究では、以上の3つのスケジューラに加えて以下のようなスケジューラが定義されている [2].

- RSYNCH : 全てのロボットが FSYNCH での動作を無限に繰り返す。あるいは FSYNCH での動作を有限回繰り返した後 (FSYNCH フェーズ), あるラウンドからは現在のラウンド  $i$  で動作するロボットとその次のラウンド  $i+1$  で動作するロボットが互いに素であるような動作に移行する (Disjoint フェーズ). なお、各ラウンドで

動作するロボットは、それぞれ完全に同期して Look, Compute, Move 動作を実行する。

## 5 モデルの表記

以下、スケジューラ  $S$  上で動作するライトモデル  $M$  のロボットを、 $M^S$  と表記する。RSYNCH スケジューラ上で動作する FCOM ロボットは  $FCOM^{RSYNCH}$ , RSYNCH スケジューラ上で動作する LUMI ロボットは  $LUMI^{RSYNCH}$  と表記される。

## 6 モデル間の計算能力の関係

先行研究により、ロボットのモデル間の計算能力には図1のような関係があることが分かっている。なお、図において、 $A > B$  はモデル  $B$  よりもモデル  $A$  の方が計算能力が高い (モデル  $B$  で解ける問題の集合がモデル  $A$  で解ける問題の真部分集合になっている) こと、 $A \equiv B$  はモデル  $A$  とモデル  $B$  の計算能力が等価である (モデル  $B$  で解ける問題の集合とモデル  $A$  で解ける問題の集合が等しい) こと、 $A \perp B$  はモデル  $A$  とモデル  $B$  の計算能力が比較できないこと (モデル  $A$  では解けないがモデル  $B$  で解ける問題が存在し、かつモデル  $B$  では解けないがモデル  $A$  で解ける問題が存在すること) を示している。本研究では、特に  $FCOM^{RSYNCH}$  と  $LUMI^{RSYNCH}$  との間の計算能力の関係に注目している。

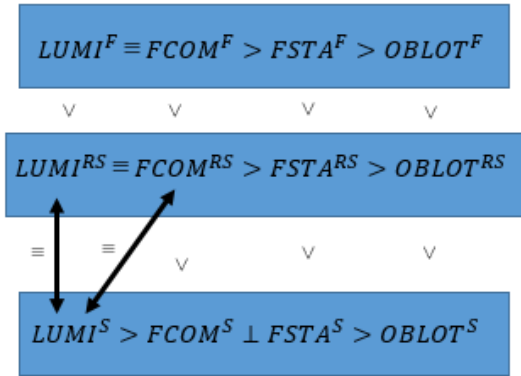


図 1: ロボットのモデル間の計算能力の関係

## 7 ロボットのシミュレーション

### 7.1 FCOM ロボットによる LUMI ロボットのシミュレーション

FCOM ロボットは自身の持つライトの色を観察することができないため、FCOM ロボットで LUMI の

動作をシミュレートする場合には、予め他のロボットに自身のライトの色をコピーしてもらい、アルゴリズム A を実行する前に自身の持つライトの現在の色を覚えてもらう、という動作が必要になる。なお、本研究で考えるロボットは平面上の点と考えているため、同じ地点に 2 台以上のロボットが存在しうる。そのため suc の位置にいるロボットのライトをコピーしてライト Suc に格納する場合、複数台のロボットが suc の位置にいることを考え、Suc には  $2^L$  色が必要になる。ここで、同一地点に複数のロボットが存在する場合、それらを観察した場合はロボットが持っている現在のライトの色の集合がみえる (set-view) ことを仮定している [3]。またシミュレーションを実行した全てのロボットが、スケジューラの条件を満足するような動作をするようにシミュレーションアルゴリズムの設計するよう注意する必要がある。

## 7.2 FSYNCH スケジューラ上でのシミュレーション

RSYNCH の部分モデルである FSYNCH スケジューラ上で FCOM ロボットによる LUMI ロボットのシミュレーションを行う場合には、すべてのロボットが相手のライトの色をコピーする処理 (Copy) と、コピーされたライトをもとにアルゴリズム A を実行する処理 (Execute) を繰り返すことでシミュレーションを実行することができる [5]。

## 8 従来のシミュレーションアルゴリズム

従来の方法では、以下に示す  $FCOM^{RSYNCH}$  による  $LUMI^{SSYNCH}$  のシミュレーションアルゴリズムと、 $LUMI^{SSYNCH}$  による  $LUMI^{RSYNCH}$  のシミュレーションアルゴリズムを組み合わせることで、間接的に  $FCOM^{RSYNCH}$  による  $LUMI^{RSYNCH}$  のシミュレーションができることが分かっている。このシミュレーションアルゴリズムは、FSYNCH でのシミュレーションの Copy, Execute に、ライトのリセットを行う処理 (Reset) が追加されたアルゴリズムになっている。

### 8.1 $FCOM^{RSYNCH}$ による $LUMI^{SSYNCH}$ のシミュレーション

先行研究 [2] により、以下の定理が明らかになっている。

**Theorem 1.** アルゴリズム A に用いるライトの色数を  $L$  とおくと、 $64 * L * 2^L$  色を持つシミュ

レーションアルゴリズムで、 $FCOM^{RSYNCH}$  による  $LUMI^{SSYNCH}$  のシミュレーションができる。

先行研究のアルゴリズムは、

1. CopyColors&Execution : アルゴリズム A に用いる相手のライトと相手のアルゴリズム A の実行フラグをコピーする step
2. PerformSimulation : アルゴリズム A を実行すべきかどうかを確認し、実行すべきであればアルゴリズム A を実行する step
3. ResetCheckingFlags : 1 で用いるチェックフラグをリセットする step
4. ResetExecutionFlags : アルゴリズム A の実行フラグをリセットする step

の 4 つの step からなり、activate したロボットは観察した相手のライトから現在実行すべき step を決定し、観察した相手のライトに応じた処理を実行していく。全てのロボットはアルゴリズム A を 1 回実行した後フラグをリセットして 1 のステップに戻る。以降、全てのロボットが step 1 で、かつフラグがリセットされている状態から全てのロボットがアルゴリズム A を実行し、フラグをリセットして 1 のステップに戻るまでの 1 サイクルをメガサイクルと呼ぶことにする。スケジューラの制約から、全てのロボットはメガサイクル内で 1 回だけアルゴリズム A を実行する。

シミュレーションアルゴリズムでは、ロボットはアルゴリズム A に用いる自身のライト light, アルゴリズム A に用いる相手のライト suc.light の他に、以下に示すライトを保持してシミュレーションアルゴリズムを実行する。

1.  $step = \{1, 2, 3, m\}$  : シミュレーションアルゴリズムのステップを表すライトで、1, 2, 3, m の 4 色である (初期状態は 1)、各 step は、それぞれ CopyColors&Execution, PerformSimulation, ResetCheckingFlags, ResetExecutionFlags を表す。
2.  $executed = \{True, False\}$  : このライトを持つロボット自身が現在のメガサイクルでアルゴリズム A を実行したか否かを示す、初期状態は False である。

3.  $\text{suc.executed} = 2^{\text{True, False}}$  : 相手のロボットが現在のメガサイクルでアルゴリズム A を実行したか否かを示す, 初期状態は  $\phi$  である.
4.  $\text{me.checked} = \{\text{True, False}\}$  : このライトを持つロボット自身がアルゴリズム A に用いる相手のライトと相手のアルゴリズム A の実行フラグをコピーしたかどうかを示す. 初期状態は False である.

## 8.2 $LUMI^{SSYNCH}$ による $LUMI^{RSYNCH}$ のシミュレーション

先行研究 [4] により, 以下の定理が明らかになっている.

**Theorem 2.** アルゴリズム A に用いるライトの色数を  $L$  とおくと,  $4 * L$  色を持つシミュレーションアルゴリズムで,  $LUMI^{SSYNCH}$  による  $LUMI^{RSYNCH}$  のシミュレーションができる.

先行研究のアルゴリズムは,

1. T(rying): まだアルゴリズム A を実行していない
2. M(oving): アルゴリズム A を実行した
3. S(topped): アルゴリズム A を実行済だが, まだアルゴリズム A を実行していないロボットが存在している (最後ではない)
4. S' (topped): アルゴリズム A を最後に実行したの 4 状態を表すライト (初期状態 T) を用いる.

## 8.3 $FCOM^{RSYNCH}$ による $LUMI^{RSYNCH}$ のシミュレーション (従来の方法)

Theorem1, 2 より, 以下の定理が導かれる.

**Theorem 3.** アルゴリズム A に用いるライトの色数を  $k$  とおくと,  $64 * 4L * 2^{4L}$  色を持つシミュレーションアルゴリズムで,  $FCOM^{RSYNCH}$  による  $LUMI^{RSYNCH}$  のシミュレーションができる.

従来の方法では 2 つのシミュレーションアルゴリズムを組み合わせている関係からシミュレーションに必要なライトの色数が多くなっている. そこで本研究では,  $FCOM^{RSYNCH}$  で直接  $LUMI^{RSYNCH}$  をシミュレーションするアルゴリズムを考えることで, 必要なライトの色数を減少させることが考えたと予想し, アルゴリズムの設計を行った.

## 8.4 ロボットの台数を 2 台に制限した場合

なお, ロボットの台数を 2 台に制限すると,  $FCOM^{RSYNCH}$  による  $LUMI^{RSYNCH}$  のシミュレーションに必要な状態数を大幅に削減することができる [6].

**Theorem 4.** ロボットの台数が 2 台のとき, アルゴリズム A に用いるライトの色数を  $L$  とおくと,  $3 * L * 2^L$  色を持つシミュレーションアルゴリズムで,  $FCOM^{RSYNCH}$  による  $LUMI^{RSYNCH}$  のシミュレーションができる.

## 9 $FCOM^{RSYNCH}$ による $LUMI^{RSYNCH}$ の直接シミュレーション

本研究では, 先行研究 [4] で提案されている  $LUMI^{SSYNCH}$  による  $LUMI^{RSYNCH}$  のシミュレーションアルゴリズムを改良することで,  $FCOM^{RSYNCH}$  による  $LUMI^{RSYNCH}$  の直接シミュレーションができるアルゴリズムの作成を行った. 今回作成したシミュレーションアルゴリズムの概要は以下の通りである. 以下, 提案したシミュレーションアルゴリズムをアルゴリズム Sim と呼ぶ.

### 9.1 アルゴリズム Sim の概要

アルゴリズム Sim は, 自身の  $\text{suc}$  の位置にいるロボットのライトをコピーする Copy と, 相手のライトおよび予め自身の  $\text{pred}$  にコピーさせた自身のライトの色を元に先行研究 [4] での  $LUMI^{SSYNCH}$  による  $LUMI^{RSYNCH}$  のシミュレーションアルゴリズムを実行する Execute, ライトのリセットの準備をする SetSucExe, そして次の Copy のためにライトをリセットする Reset の 4 つの処理からなる.

各ロボット  $r$  は, シミュレーションしたいアルゴリズムに用いるライト  $r.\text{Alight}$ , そして自身の  $\text{suc}$  の  $r.\text{Alight}$  を格納するための  $r.\text{Suc.Alight}$  の他に, T, M, S, S' の 4 色を持つ  $r.\text{Executed}$  と,  $2^{T, M, S, S'}$  を持つ  $r.\text{Suc.Executed}$  の 2 種類のライトを用いる.  $r.\text{Executed}$  は先行研究 [4] でのシミュレーションアルゴリズムを実行するためのライトであり, 先行研究と全く同じ用途で利用される.  $r.\text{Suc.Executed}$  は Look で直接観察することができない自身の  $r.\text{Executed}$  を格納するためのライトであり, また現在自身が Copy と Executed のどちらを実行するかを判断するためにも用いられる.

初期状態では, 全てのロボットは  $r.\text{Executed} = \text{T}$ ,  $r.\text{Suc.Executed} = \phi$  となっている. アクティ

ベートしたロボットは自身以外の全てのロボットのライトを観察し、 $r.Suc.Executed = \phi$ であるロボットが存在する場合は Copy を実行する。つまり  $r.Suc.Align$  に自身の  $suc$  の位置にいるロボットの  $r.Align$ ,  $r.Suc.Executed$  に自身の  $suc$  の位置にいるロボットの  $r.Executed$  を格納する。ラウンドが進んでいくと、やがてアクティベートしたロボットは観察した全てのロボットの  $r.Suc.Executed$  に  $suc$  のライトが格納されている状況を観察することになる。このとき、自身はまだライトのコピーを終えていない可能性があることに注意が必要である。観察した全てのロボットの  $r.Suc.Executed$  に  $suc$  のライトが格納されている状況を観察したロボットは Execute を実行する。具体的にはまず Copy を実行し、先行研究 [4] でのシミュレーションアルゴリズムを実行して自身の  $r.Executed$  の色を必要に応じて変化させる。そして自身の  $r.Suc.Executed$  に T, M, S, S' を格納する。

あるロボットが Execute を実行した次のラウンドからは、変化したライトのコピーが必要になるのでコピーの準備に移行する。あるロボットが Execute を実行した次のラウンドにアクティベートしたロボットは、 $Suc.Executed$  が T, M, S, S' であるロボットがいることを観察して現在がコピーの準備を行う状況であると判断し、SetSucExe を行う。具体的には自身の  $r.Suc.Executed$  を T, M, S, S' にする。その後はアクティベートしたロボットは観察した全てのロボットの  $r.Suc.Executed$  が T, M, S, S' となるまで SetSucExe を行う。

ラウンドが進んでいくと、やがてアクティベートしたロボットは全てのロボットの  $r.Suc.Executed$  が T, M, S, S' であることを観察することになる。この状況を観察したロボットは Reset を実行する。具体的には自身の  $r.Suc.Align$  と  $r.Suc.Executed$  をいずれも  $\phi$  にする。以降アクティベートしたロボットは、観察した全てのロボットの  $r.Suc.Align$  と  $r.Suc.Executed$  がいずれも  $\phi$  となるまで、Reset を実行する。

ラウンドが進んでいくと、やがてアクティベートしたロボットは全てのロボットの  $r.Suc.Align$  と  $r.Suc.Executed$  がいずれも  $\phi$  であることを観察することになる。自身は Reset を実行していない可能性があるため Reset を実行し、初期状況と同様にして再び Copy を実行する。

アルゴリズム Sim での  $r.Suc.Align$  および  $r.Suc.Executed$  の状態遷移図を図 2 に示す。図中

の赤矢印は FSYNCH フェーズ、青矢印は Disjoint フェーズでの遷移を表す。

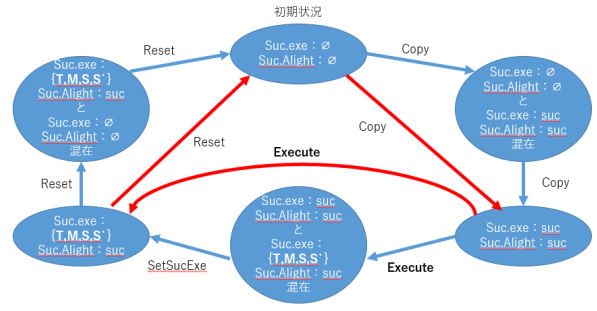


図 2:  $r.Suc.Align$  および  $r.Suc.Executed$  の状態遷移図（頂点は観察した相手の Suc, 矢印は実行する処理と自身の Suc の変化を表す）

## 10 結論と今後の展望

本研究では、RSYNCH と呼ばれる特殊なスケジューラ上で、2 台以上の FCOM ロボットで同数の LUMI ロボットの動作をシミュレーションできるようなアルゴリズムについて検討した。従来の方法で  $m$  用いられていたは 2 つのアルゴリズム統合することでシミュレーションを効率化し、必要なライトの色数を  $64 * 4L * 2^{4L}$  色から  $60 * L * 2^L$  色に削減することができた。今回取り組んだテーマの他に、シミュレーションに必要なライトの色数の更なる削減や下界の調査といった課題が残されている。

## 参考文献

- [1] P. Flocchini, G. Prencipe, and N. Santoro. Distributed Computing by Oblivious Mobile Robots. Morgan & Claypool, 2012.
- [2] K. Buchin, P. Flocchini, I. Kostitsyn, T. Peters, N. Santoro, K. Wada, On the Computational Power of Energy-Constrained Mobile Robots: Algorithms and Cross-Model Analysis, 29th International Colloquium on Structural Information and Communication Complexity (SIROCCO 2022), LNCS 13298, (Jun. 2022) 42-61.
- [3] S. Terai and K. Wada and Y. Katayama Gathering problems for autonomous mobile robots with lights (tcs 941, 241–261, 2023)
- [4] K. Nakajima, K. Takase, K. Wada: Effi-

- cient Self-stabilizing Simulations of Energy-Restricted Mobile Robots by Asynchronous Luminous Mobile Robots, 26th SSS (Accepted), ArXiv.2403.05542, (2024).
- [5] Paola Flocchini, Nicola Santoro and Koichi Wada. On Memory, Communication, and Synchronous Schedulers When Moving and Computing, 23rd International Conference on Principles of Distributed Systems (OPODIS 2019), 25:1–25:17.
- [6] 小原直将, 和田幸一 2 台の FCOM ロボットによる LUMI ロボットのシミュレーション可能なスケジューラについて : 第 18 回情報科学ワークショップ (Sep. 2022), 1-3.4.