Fast Uniform Scattering on a Grid for Asynchronous Oblivious Robots

Pavan Poudel and Gokarna Sharma(✉)

Department of Computer Science, Kent State University, Kent, OH 44242, USA
{ppoudel,sharma}@cs.kent.edu

Abstract. We consider $K = (k+1) \times (k+1)$ autonomous mobile robots operating on an anonymous $N = (n + 1) \times (n + 1)$-node grid, $n = k \cdot d, d \geq 2, k \geq 2$, following *Look-Compute-Move* cycles under the *classic oblivious robots* model. Starting from any initial configuration of robots positioned on distinct grid nodes, we consider the *uniform scattering* problem of repositioning them on the grid nodes so that each robot reaches to a static configuration in which they cover uniformly the grid. In this paper, we provide the first $O(n)$ time, collision-free algorithm for this problem in the asynchronous setting, given that the robots have common orientation, knowledge of n and k, $O(1)$-bits of memory, and visibility range of $2 \cdot \max\{n/k, k\}$. The best previously known algorithm for this problem on a grid has runtime $O(n^2/d)$ (or $O(nk)$) with the same robot capabilities in the asynchronous setting except the visibility range $2 \cdot n/k$. The proposed algorithm is asymptotically time-optimal since there is a time lower bound of $\Omega(n)$.

1 Introduction

The well-studied model in distributed computing by a team of autonomous mobile robots is the *classic oblivious robots* (COR) model [8] where the robots in the team are *points* (do not occupy any space), *autonomous* (no external control), *anonymous* (no unique identifiers), *indistinguishable* (no external identifiers), *disoriented* (no agreement on coordinate systems and units of distance measures), *oblivious* (no memory of past computation), and *silent* (no direct communication and actions are coordinated via only vision and mobility). The robots operate on a plane and execute the same algorithm. The robots perform their computation in *Look-Compute-Move* (LCM) cycles: an active robot first gets a snapshot of its surroundings (*Look*), computes a destination point based on the snapshot (*Compute*), and finally moves to the destination point (*Move*).

In this paper, we assume that robots operate on a grid G. We assume that nodes and edges of G are *unlabeled*, i.e., robots cannot differentiate one node (edge) from another. The robots reside at nodes of G and they can move from one node to another following the edges of G. The non-neighbor nodes in G are visited following the intermediate nodes of G. We assume that, if a robot at a

S. Devismes and N. Mittal (Eds.): SSS 2020, LNCS 12514, pp. 211–228, 2020.
https://doi.org/10.1007/978-3-030-64348-5_17

Table 1. Results for Uniform Scattering on a grid.

Algorithm	Model	Visibility range	Runtime	Setting
Barriere *et al.* [2]	Classic oblivious	$2 \cdot n/k$	$O(nk)$	Asynchronous
Poudel and Sharma [18]	Robots with lights	$2 \cdot n/k$	$\Theta(n)$	Fully Synchronous
Theorem 1	**Classic oblivious**	$2 \cdot \max\{n/k, k\}$	$\Theta(n)$	Asynchronous

node computes a destination point (to move) in one LCM cycle, then that destination point is the neighboring node of the node where that robot is currently positioned.

We study the fundamental Uniform Scattering problem on an anonymous (square) grid G of $N = (n+1) \times (n+1)$ nodes for a set of $K = (k+1) \times (k+1)$ robots, which is defined as follows: Given any initial configuration of K robots positioned on distinct nodes of G, the robots reposition to reach a configuration in which each robot is on a distinct node of G and they uniformly cover G (see Fig. 1).

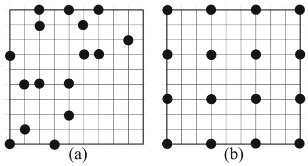


Fig. 1. (a) Initial configuration; (b) Uniform Scattering.

This problem has practical applications when a team of randomly deployed robots in a region have to cover the region uniformly to maximize the coverage for different purposes, such as intruder detection. An essential requirement is clearly that the robots will reach a state of *static equilibrium* and that scattering is completed as fast as possible. It is assumed that $n = k \cdot d$, $d \geq 2$, $k \geq 2$ to guarantee a final Uniform Scattering configuration.

Barriere *et al.* [2] studied Uniform Scattering for the first time in the COR model, providing a deterministic algorithm in the asynchronous setting given that the robots have the following capabilities:

- *common orientation* – each robot has consistent notion of North-South and West-East, e.g., as provided by a compass,
- *knowledge* of parameters n and k,
- a *visibility range* of $2 \cdot \lfloor n/k \rfloor$ (i.e., a robot can see robots within distance $2 \cdot \lfloor n/k \rfloor$),
- $O(1)$-*bits of memory* in each robot to store the different states of the system.

Barriere *et al.* [2] did not formally analyze the runtime; however, it is easy to show that their algorithm has runtime $O(n^2/d)$ (or $O(nk)$). Recently, Poudel and Sharma [18] proposed a $\Theta(n)$-time algorithm in the fully synchronous setting for Uniform Scattering in the *robots with lights* (RWL) model [4], where robots have an externally visible light that can assume a distinct color at a time from a given constant sized set. In this paper, our goal is to design a faster algorithm in the asynchronous setting for Uniform Scattering in the COR model (see Table 1 for the comparison).

Contributions. We consider the robots and problem setting on a grid as in Barriere *et al.* [2], except the visibility range $2 \cdot \max\{\lfloor n/k \rfloor, k\}$. *Unobstructed visibility* is considered where a robot sees all other robots within its visibility range. *Asynchronous* setting is considered where the robots perform their LCM cycles at arbitrary times. Two robots cannot move to the same node in G. This would constitute a *collision*. We prove:

Theorem 1. *For any initial configuration of* $K = (k+1) \times (k+1)$ *robots positioned on distinct nodes of an anonymous square grid* G *of* $N = (n+1) \times (n+1)$ *nodes with each robot having the visibility range* $2 \cdot \max\{n/k, k\}$*,* Uniform Scattering *can be solved in* $\Theta(n)$ *time in the asynchronous setting avoiding collisions, when robots have common orientation, knowledge of* n *and* k*, and* $O(1)$ *bits of internal memory.*

Theorem 1 improves significantly on the $O(nk)$ (or $O(n^2/d)$) runtime of Barriere *et al.* [2] for the COR model under the same capabilities, except that our algorithm has visibility range $2 \cdot \max\{n/k, k\}$ whereas Barriere *et al.* [2] has $2 \cdot n/k$. Interestingly when $k \leq \sqrt{n}$, our visibility range matches with the visibility range of Barriere *et al.*

Techniques. The time lower bound can be established by showing the minimum number of times some robot has to move to reach a Uniform Scattering configuration. The time lower bound established in Poudel and Sharma [18] immediately proves this lower bound. For the time upper bound, we provide a deterministic algorithm that works in three phases, Phase 1 to Phase 3, executed sequentially.

- **Phase 1 (Gather):** In this phase, all K robots are repositioned on the distinct nodes in the top-right part of G forming a square sub-grid G' (which we call the *gathering configuration* C_{gather}; formal definition in Sect. 2). Essentially what happens in C_{gather} is that robots occupy the $(k+1) \times (k+1)$ sub-grid G', one robot on each node of G'. C_{gather} is obtained through two kinds of moves: (i) Northeast moves and (ii) Balancing moves. A robot performs Northeast moves to reach G'. The robot moves either vertically North or horizontally East during Northeast moves. After reaching G', the robot switches to Balancing moves. In the Balancing moves, based on the configuration of other robots, the robot may move North, East, South or West inside G', facilitating new incoming robots to be accommodated inside G' fast. We show that this process results C_{gather} in $O(n)$ time.
- **Phase 2 (Pre-scatter):** The robots in C_{gather} move horizontally West to occupy $(k+1)$ columns of G with distance between two subsequent columns exactly $d = n/k$. In each such column, the $k+1$ robots occupy $k+1$ consecutive positions from the top boundary line of G, which we call the pre-scatter configuration $C_{pre-scatter}$. In this phase, when a robot sees at least one robot on the same horizontal line in the East at distance less than d and the neighboring node in the West is empty, it moves to the West. We show that this phase finishes in $O(n)$ time.

– **Phase 3 (Scatter):** The $k+1$ robots in each of the $k+1$ columns move vertically South maintaining a fixed distance $d = n/k$ between consecutive robots. There are $k+1$ final positions on each line, and hence a Uniform Scattering configuration is achieved when robots move to the final positions on those lines. The algorithm then terminates. We will show that this phase also finishes in $O(n)$ time.

Therefore, the overall runtime of the algorithm becomes $O(n)$, which is asymptotically optimal given the time lower bound of $\Omega(n)$. Executing Phases 1–3 becomes relatively straightforward in the fully synchronous setting. The challenge is on how to execute Phases 1–3 correctly in the asynchronous setting. For simplicity in understanding, we present the synchronous case first and extend it later to the asynchronous case.

Related Work. Uniform Scattering is the subject of extensive research in several fields. The research literature is vast and we only discuss in brief the aspects related to our work. In cooperative mobile swarm robotics, this question has been studied in terms of *scattering*, *coverage*, and a special case of *formation* [1,3,5,9,10,13,15,22]. Uniform Scattering has also been studied in terms of *self-deployment* in mobile sensor networks and in networks of robotic sensors [11,12,16,17,19,21].

The existing works differ on whether robots (sensors) operate on plane or on graphs. They also differ on various parameters, e.g., (i) synchronization settings (fully synchronous, semi-synchronous, or asynchronous), (ii) the robots are oblivious or have persistent memory, (iii) unlimited or limited visibility range, (iv) exact or approximate covering, (v) termination guarantees, (vi) knowledge of the number of robots in the system, (vii) obstructed/unobstructed visibility, (viii) knowledge of the global coordinate system/common orientation/chirality/one-axis agreement, etc.

Our work is on graphs, particularly grids. The grid setting was heavily used in self-deployment and covering problems. We build upon the only previous work of Barriere *et al.* [2] in the COR model. Furthermore, scattering is considered in [14] in the Euclidean plane under limited visibility and non-trivial time bounds (lower and upper) were reported. Scattering on a ring is considered in [6,7,20].

Paper Organization. We discuss model and some preliminaries in Sect. 2. The algorithm in the fully synchronous setting is presented in Sect. 3 and the extension to the asynchronous setting is provided in Sect. 4. Finally, we conclude in Sect. 5. Some proofs and pseudocodes are omitted due to space constraints.

2 Model and Preliminaries

Graph. Let $G = (V, E)$ be a grid of $N = (n+1) \times (n+1)$ nodes, where $V = \{v_1, v_2, \ldots\}$ denotes the node sets and $E \subseteq V \times V$ denotes the edge sets. Each node v_i represents a point location and each edge $(v_i, v_j), i \neq j$, represents a line connecting any two nodes v_i and v_j of V. We assume that the grid is anonymous, i.e., nodes and edges of G are unlabeled. We assume that each edge of G is of unit distance length.

Robots. Let $\mathcal{R} = \{r_1, r_2, \ldots, r_K\}$ be a set of $K = (k+1) \times (k+1)$ robots residing on the nodes of grid G. No robot can reside on the edges of G at any time (except in motion). Moreover, no two robots can occupy the same node of G. In the initial configuration, we assume that robots in $\mathcal{R}$ are at distinct nodes of G and maintain this property throughout the execution of the algorithm. In the algorithm description, we denote by r_i the robot r_i and v_i the node on which r_i resides. The robots have the visibility range $2 \cdot \max\{n/k, k\}$. We assume unobstructed visibility, i.e., a robot sees all other robots within distance $2 \cdot \max\{n/k, k\}$ (even if the robots are collinear). Following Barriere *et al.* [2], we assume that robots can detect the boundary lines of G when they are $\leq 2 \cdot \max\{n/k, k\}$ distance away from the boundary of G.

Common Orientation. The common orientation means that each robot has a consistent notion of "North-South" and "West-East", e.g., as provided by a compass [2]. For the common orientation, no access to any global localization system is required, i.e., the robots do not need to know their own position on the grid G. We assume that the edges of G are consistently labeled $North$ (top), $South$ (bottom), $West$ (left), and $East$ (right), and edge labels are visible to robots.

Look-Compute-Move. At any time, a robot $r_i \in \mathcal{R}$ could be active or inactive. When a robot r_i becomes active, it performs the "Look-Compute-Move" cycle as follows.

– *Look:* For each robot r_j that is visible to it, r_i can observe the position of r_j on G. Robot r_i can also know its own position.
– *Compute:* In any LCM cycle, r_i may perform an arbitrary computation using only the positions observed during the "look" portion of that cycle. This includes determination of a (possibly) new position (which is a node of G) and internal memory storage for r_i for the start of the next cycle. Robot r_i maintains this new memory information from that cycle to the next.
– *Move:* At the end of the LCM cycle, r_i changes its memory to the new information and moves to its new position.

Robot Activation and Time. In the fully synchronous setting ($\mathcal{F}SYNC$), every robot is active in every LCM cycle. In the semi-synchronous setting ($\mathcal{S}SYNC$), at least one robot is active, and over an infinite number of LCM cycles, every robot is active infinitely often. In the asynchronous setting ($\mathcal{A}SYNC$), there is no common notion of time and no assumption is made on the number and frequency of LCM cycles in which a robot can be active; nevertheless, each robot is active infinitely often. For the $\mathcal{F}SYNC$, time is measured in *rounds*. For the $\mathcal{S}SYNC$ and $\mathcal{A}SYNC$, time is measured in epoch. An *epoch* is the smallest interval of time within which each robot is guaranteed to be active at least once.

Configuration. A *configuration* $C_t = \{(r_1^t, mem_1^t), \ldots, (r_K^t, mem_K^t)\}$ defines the positions of the robots in $\mathcal{R}$ on the nodes of G and their internal memory for any time $t \geq 0$. A configuration for a robot $r_i \in \mathcal{R}$, $C_t(r_i)$, defines the positions of the robots in $\mathcal{R}$ that are visible to r_i (including r_i) and their memory, i.e., $C_t(r_i) \subseteq C_t$, at time t. Since each robot has visibility range $2 \cdot \max\{n/k, k\}$, $C_t(r_i)$ has the robots that are within distance $2 \cdot \max\{n/k, k\}$ from r_i. For simplicity and clarity, we sometime write $C, C(r_i)$ to denote $C_t, C_t(r_i)$, respectively. The configuration C_t at $t = 0$ is called the *initial configuration* C_{init}, in which K robots are on K distinct nodes of G.

Uniform Scattering. *Given an anonymous grid* $G = (V, E)$ *of* $N = (n+1) \times (n+1)$ *nodes and a team of* $K = (k+1) \times (k+1)$ *robots with* $n = k \cdot d, k \geq 2, d \geq 2$, *positioned initially arbitrarily on the distinct nodes of* G*, reposition the robots autonomously to reach an equilibrium such that the nodes* $(i \cdot d, j \cdot d)$ *of* G *with* $i, j \in [0, k]$ *hosting exactly one robot each.* We say nodes $(i \cdot d, j \cdot d)$ with $i, j \in [0, k]$ the *final* positions. We say a node (x, y) of G *occupied* (or *non-empty*), if there is a robot positioned on it.

Gathering Configuration. Let $\mathcal{R}$ be a set of K robots positioned on the distinct nodes of G. Let L_N, L_S, L_W, L_E be the North, South, East and West boundary lines of G, respectively. Let L'_W and L'_S be the vertical and horizontal lines parallel to L_E and L_N and passing through k hops West and South of L_E and L_N, respectively. Let G' be the sub-grid of G enclosed by lines L_E, L_N, L'_W, and L'_S (including the nodes of G on L_E, L_N, L'_W, L'_S) in G. We say that a robot $r_i \in \mathcal{R}$ is in a gathering configuration C_{gather} if r_i lies on G' and r_i sees all the nodes in G' are occupied (Fig. 2). We say that the robots in the set $\mathcal{R}$ are in C_{gather}, if each robot in $\mathcal{R}$ is in C_{gather}. Therefore, in C_{gather}, the $(k+1) \times (k+1)$ sub-grid on the topright part of G is occupied with robots. Moreover, we define two regions w.r.t. G'. The grid area of G in the West of G' between L_N and L'_S is denoted as *west-region* of G'. The grid area of G in the South of G' between L_E and L'_W is denoted as *south-region* of G'.

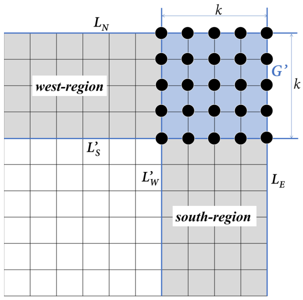


Fig. 2. C_{gather}.

3 Uniform Scattering Algorithm in $\mathcal{FSYNC}$

We now describe our collision-free, time-optimal $O(n)$-round Uniform Scattering algorithm in the $\mathcal{FSYNC}$ setting. The pseudocode is given in Algorithm 1. The robots have the common orientation, knowledge of parameters n and k, visibility range of $2 \cdot \max\{\lfloor n/k \rfloor, k\}$, and $O(1)$-bits of memory internal to each robot. We describe the algorithm with respect to a single robot $r_i \in \mathcal{R}$. Figure 3 depicts what intuitively Phases 1–3 do to solve Uniform Scattering starting from any arbitrary C_{init}.

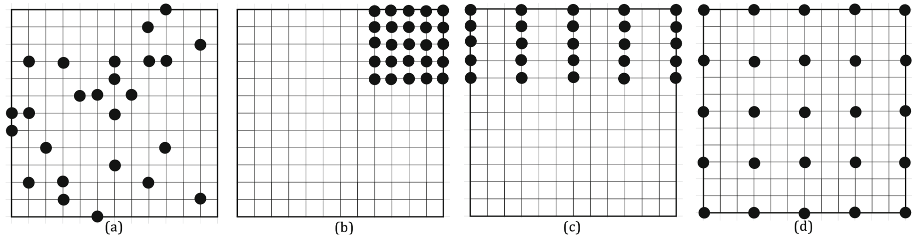


Fig. 3. (a) Initial configuration C_{init}; (b) Gathering configuration C_{gather} (Phase 1); (c) Pre-scatter Configuration (Phase 2); (d) Uniform Scattering Configuration (Phase 3).

Algorithm 1: UNIFORM_SCATTER(r_i, n, k, G)

$C(r_i) \leftarrow$ configuration C for robot r_i (including r_i);
$L_E, L_W, L_N, L_S \leftarrow$ East, West, North and South boundary lines of G, respectively;
$L'_W \leftarrow$ vertical line parallel to L_E at distance k west from L_E;
$L'_S \leftarrow$ horizontal line parallel to L_N at distance k south from L_N;
$G' \leftarrow$ subgraph of G enclosed by lines L_E, L_N, L'_W and L'_S;
$H(r_i), V(r_i) \leftarrow$ horizontal and vertical lines on G passing through r_i, respectively;
$r_i \cdot state \leftarrow 0$ (initial state of r_i); $d \leftarrow n/k$;
if $r_i \cdot state = 0$ **then** $GATHER(r_i, H(r_i), V(r_i), L_E, L_N, L'_S, L'_W, G', C(r_i))$;
else if $r_i \cdot state = 1$ **then** $PRE\_SCATTER(r_i, d, H(r_i), V(r_i), L_E, L_W, L'_S, C(r_i))$;
else if $r_i \cdot state = 2$ **then** $SCATTER(r_i, d, V(r_i), L_N, L_S, C(r_i))$;

Algorithm 2: GATHER$(r_i, H(r_i), V(r_i), L_E, L_N, L'_S, L'_W, G', C(r_i))$

$(x_i, y_i) \leftarrow$ current position of r_i in G;
if $(x_i, y_i) \in G'$ **then**
  if r_i sees all the nodes of G' occupied **then** $r_i \cdot state \leftarrow 1$;
  else $BALANCE(r_i, H(r_i), V(r_i), L_E, L_N, L'_W, L'_S, G', C(r_i))$;
else if $(x_i, y_i + 1)$ is empty $\wedge$ $((x_i, y_i + 1) \notin G' \vee ((x_i, y_i + 1) \in G' \wedge (x_i + 1, y_i + 1)$ is empty $\wedge$ r_i sees no robot in the West of G' between L_N and $L'_S))$ **then**
  r_i moves to $(x_i, y_i + 1)$;
else if $(x_i + 1, y_i)$ is empty $\wedge$ $(((x_i + 1, y_i) \in G' \wedge (x_i + 1, y_i + 1)$ is empty$)$ $\vee((x_i + 1, y_i) \notin G' \wedge (x_i + 1, y_i - 1)$ is empty$))$ **then** r_i moves to $(x_i + 1, y_i)$;

Phase 1 (Gather). The purpose of Phase 1 is to reach a gathering configuration C_{gather} starting from C_{init} (Fig. 3(a)–(b)). The pseudocode is given in Algorithm 2. Phase 1 has two sub-phases, Phase 1.1 (Northeast moves) and Phase 1.2 (Balancing moves), which execute sequentially one after another. In Phase 1.1, robots move towards the North-East of G until they reach G'. After a robot reaches G', it switches to Phase 1.2 doing balancing moves to reposition itself inside G'. We guarantee that after a robot enters G', it never moves out of G' during Phase 1. By the end of Phase 1, all K robots are positioned on the distinct nodes of G' achieving C_{gather}. We will prove that Phases 1.1 and 1.2 each run for $O(n)$ rounds. We describe Phase 1.1 and 1.2 in detail below.

Phase 1.1 (Northeast Moves). Let (x_i, y_i) be the current position of robot r_i in G. In Phase 1.1, r_i does the following in each LCM cycle.

Algorithm 3: BALANCE($r_i, H(r_i), V(r_i), , L_E, L_N, L'_W, L'_S, G', C(r_i)$)

if r_i sees no robot in the South of G' between L_E and $L'_W \vee r_i$ sees at least a robot at distance $\leq (2k-2)$ in the West of G' between L_N and L'_S **then** $MoveSE()$;
else if r_i sees no robot in the West of G' between L_N and $L'_S \wedge r_i$ sees at least a robot at distance $\leq (2k-2)$ in the South of G' between L_E and L'_W **then** $MoveWN()$;

Algorithm 4: MoveSE()

$r_{south} \leftarrow$ southmost robot seen by r_i in the West of G';
$L_{ref} \leftarrow$ horizontal reference line passing through r_{south};
$d_{ref} \leftarrow$ distance between L_N and L_{ref};
$dx, dy \leftarrow$ distance from r_i to L'_W and L_{ref}, respectively;
if $(x_i, y_i - 1)$ is empty $\wedge$ $dx \geq dy$ $\wedge$ there exist less than $(k - d_{ref})$ robots on $V(r_i)$ in the South of r_i **then** r_i moves to $(x_i, y_i - 1)$;
else if $(x_i + 1, y_i)$ and $(x_i + 1, y_i + 1)$ are empty **then** r_i moves to $(x_i + 1, y_i)$;

- Move to $(x_i, y_i + 1)$, if that position (i.e., grid node) is empty and either:
  i. $(x_i, y_i + 1)$ does not lie on G', or
  ii. $(x_i, y_i + 1)$ lies on G', $(x_i + 1, y_i + 1)$ is empty and r_i sees no robot in the *west-region* of G'. (Note: This condition prevents possible collision of r_i with another robot r_j inside G' due to the balancing move of r_j.)
- Otherwise, move to $(x_i + 1, y_i)$, if $(x_i + 1, y_i)$ is empty, and either:
  i. $(x_i + 1, y_i)$ lies on G' and there is no robot on $(x_i + 1, y_i + 1)$, or
  ii. $(x_i + 1, y_i)$ does not lie on G' and there is no robot on $(x_i + 1, y_i - 1)$.
- Switch to Phase 1.2, if it lies on G' and G' is not fully occupied.
- Switch to Phase 2, if G' is fully occupied.

Phase 1.2 (Balancing Moves). The pseudocode for Phase 1.2 is in Algorithm 3. When a robot r_i reaches G', it performs balancing moves as follows. (Note that the robot r_i never moves outside of G' during Phase 1.2.).

Case 1 – r_i **sees no robot in the *south-region* of** G' **OR** r_i **sees at least a robot at distance** $\leq (2k-2)$ **in the *west-region* of** G'**:** r_i moves either South or East. r_i first checks for possible move towards South, and then towards East. Let (x_i, y_i) be the current position of r_i in G and $H(r_i), V(r_i)$ be the horizontal and vertical lines passing through r_i, respectively. Let r_{south} be the southmost robot seen by r_i in the *west-region* of G' and L_{ref} be the horizontal reference line passing through r_{south}. Let L'_W be the westmost vertical line of G'. Let d_{ref} be the distance between L_N and L_{ref}, dx be the distance from r_i to L'_W and dy be the distance from r_i to L_{ref}. Then, r_i moves South to $(x_i, y_i - 1)$, if the following conditions are satisfied: (i) $(x_i, y_i - 1)$ is empty, (ii) $dx \geq dy$, and (iii) r_i sees less than $(k - d_{ref})$ robots on $V(r_i)$ in the South of r_i.

Else if $(x_i + 1, y_i)$ and $(x_i + 1, y_i + 1)$ are empty, then r_i moves East to $(x_i + 1, y_i)$.

Case 2 – r_i **sees no robot in the *west-region* of** G' **but sees at least a robot at distance** $\leq (2k-2)$ **in the *south-region* of** G'**:** r_i moves either West or North. r_i first checks for possible move towards West, and then towards

Algorithm 5: MoveWN()

```
1 r_west ← westmost robot seen by r_i in the South of G';
2 L'_ref ← vertical reference line passing through r_west;
3 d'_ref ← distance between L_E and L'_ref;
4 dx', dy' ← distance from r_i to L'_ref and L'_S, respectively;
5 if (x_i − 1, y_i) is empty ∧ dy' ≥ dx' ∧ there exist less than (k − d'_ref) robots on H(r_i) in the
  West of r_i then r_i moves to (x_i − 1, y_i);
6 else if (x_i, y_i + 1) and (x_i + 1, y_i + 1) are empty then r_i moves to (x_i, y_i + 1);
```

North. Let r_{west} be the westmost robot seen by r_i in the *south-region* of G' and L'_{ref} be the vertical line passing through r_{west}. Let L'_S be the southmost horizontal line of G'. Let d'_{ref} be the distance between L_E and L'_{ref}. Let dx' and dy' be the distances from r_i to L'_{ref} and L'_S, respectively. Then, r_i moves one unit West, if the following conditions are satisfied: (i) $(x_i - 1, y_i)$ is empty, (ii) $dy' \geq dx'$, and (iii) r_i sees less than $(k - d'_{ref})$ robots on $H(r_i)$ in the West of r_i.

Else if (x_i, y_i+1) and (x_i+1, y_i+1) are empty, r_i moves North to (x_i, y_i+1).

Recall that a robot reaches Phase 1.2 after Phase 1.1; however, two different sets of robots execute Phase 1.1 and Phase 1.2 in parallel. While the robots inside G' are performing *Balancing* moves, the robots outside G' are performing *Northeast* moves. Phase 1.2 starts after at least a robot reaches G'. Phase 1.1 ends when all the robots reach Phase 1.2. When all the robots reach Phase 1.2, gathering configuration C_{gather} is achieved and Phase 1.2 also ends. That means, Phase 1.1 and 1.2 both end together.

Lemma 1. *Phase 1.2 starts in at most $O(n)$ rounds after Phase 1.1.*

Proof. If a robot r_j lies inside G' in the initial configuration C_{init}, then r_j directly reaches Phase 1.2. In this case, both Phase 1.1 and Phase 1.2 start at the same time. Let us analyze the case where no robot lies inside G' in C_{init}. Let r be the topmost and rightmost robot in the initial configuration C_{init} of K robots in G. Let Phase 1.1 starts and r executes Algorithm 2. Since, r is the topmost and rightmost robot in G, it moves North until it reaches either G', or north boundary line L_N of G. If r reaches G', it has taken less than n rounds and Phase 1.2 starts. Otherwise, r takes at most n rounds to reach L_N, and it moves East on L_N until it reaches G' in less than next n rounds. Since, r is the topmost and rightmost robot, there is no other robot that blocks the movement of r. Hence, in less than $2n$ rounds, r reaches G' and Phase 1.2 starts. □

Lemma 2. *Phase 1.1 is collision-free.*

Lemma 3. *Phase 1.2 is collision-, deadlock-, and livelock-free.*

Lemma 4. *Phase 1 finishes in $O(n)$ rounds.*

Proof. We have two sub-phases of Phase 1 (Phase 1.1 and Phase 1.2). Phase 1.2 starts after at least a robot reaches G'. Thus, the total runtime of Phase 1 can be divided into two parts: (i) time elapsed in Phase 1.1 and (ii) runtime of Phase

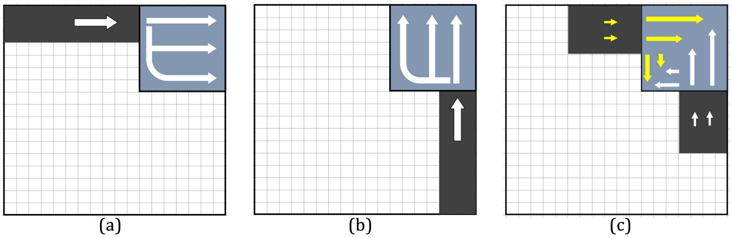


Fig. 4. Illustration of movement of robots during Phase 1; (a) if all robots reach *west-region* of G' in Phase 1.1, they move South/East inside G' in Phase 1.2; (b) if all robots reach *south-region* of G' in Phase 1.1, they move North/West inside G' in Phase 1.2; (c) if robots reach in both *west-region* and *south-region* of G' in Phase 1.1, they may perform all four types of moves (East, West, North or South) inside G' in Phase 1.2.

1.2. From Lemma 1, the time elapsed in Phase 1.1 before the start of Phase 1.2 is $O(n)$ rounds.

Now, let us analyze the runtime of Phase 1.2 with three different cases.

Case I: All robots reach the *west-region* of G' during Phase 1.1 (Fig. 4(a)). Let $R_i = \{r_i^0, r_i^1, \ldots, r_i^{p-1}\}, i = 1, 2, \ldots, n-k$, be the set of $p \leq k+1$ robots on each column at i distance west of L'_W where r_i^0 represents the robot at the northmost horizontal line, r_i^1 represents the robot on the next horizontal line below it and so on. Note that each horizontal line contains $\leq n-k$ robots and each column in the West of L'_W contains $\leq p$ robots on it. When robots in set R_1 move East, they reach G' (i.e. L'_W) and Phase 1.2 starts. In round 1 of Phase 1.2, the robots on L'_W (initially in set R_1) of G' execute Algorithm 3 to perform balancing moves. If $p = k+1$, all the robots on L'_W move East. This process repeats for all other sets of robots and it is easy to see that all the robots reach G' in $2(k+1)$ rounds. Let us analyze the scenario of $p < k+1$. In this case, the southmost robot (r_1^{p-1}) on L'_W moves South and the remaining ones move East leaving behind the top p positions on L'_W empty. During this round, the next set of robots (R_2) move East and occupy the previous positions of R_1 in the West of L'_W. In round 2 of Phase 1.2, the robots in R_2 reach L'_W, the robots which are already in G' (i.e. R_1), move further East or South (the southmost, r_1^{p-2}, moves South and others move East). This provides empty nodes for the robots currently on L'_W (i.e. R_2) to move East or South in the next round. Also, in round 2, the robots in set R_3 reach to the initial positions of R_2. In round 3, robots in R_4 reach to the initial positions of R_3, robots in R_3 reach to the initial positions of R_1 and the robots in R_2 (currently on L'_W) move East or South in G'. The robots in R_1 move further East or South by one unit. When the southmost robot r_1^{p-1} of R_1 reaches the South boundary line L'_S of G', it moves East in the next round where it meets r_1^{p-2} on its North neighboring node. In the next round, these both robots move East and meet r_1^{p-3}. Following this process, all the p robots of set R_1 ultimately reach the consecutive nodes on L_E in the South part of G'. That means, the southmost p rows of G' will be occupied by the first $k+1$ sets of robots (i.e. R_1 to R_{k+1}). Similarly, next p rows of G' will be occupied by the next $k+1$ sets of robots (i.e. R_{k+2} to R_{2k+2}). Recall that, in this case, a robot always search for a possible East or South move inside G', thus creating an empty node for each incoming robot from next column in the West. That means, in every two rounds, one column of p robots enter G'. Thus, in $2(n-k) \leq 2n$ rounds, all the robots reach G'. This achieves the gathering configuration C_{gather} and Phase 1.2 terminates.

Case II: All robots reach the *south-region* of G' during Phase 1.1 (Fig. 4(b)). This case is analogous to Case I. Here, each vertical line contains $\leq n-k$ robots on it. In this case, robots reach G' performing North move from each of the eastmost $q \leq k+1$ vertical lines. Once a robot reaches G', it performs West or North move inside G'. Following the arguments of Case I analogously, in every 2 rounds, one robot each from the q vertical lines reaches G'. That means, in $\leq 2n$ rounds, all the robots reach G' having the gathering configuration C_{gather} and Phase 1.2 terminates.

Case III: There are robots in both sides (*west-region* and *south-region*) of G' during Phase 1.1 (Fig. 4(c)). This case is the combination of Case I and Case II. In Phase 1.1, the robots in the *south-region* of G' do not move to G' until they see robots in the *west-region* of G'. That means, first all the robots in the *west-region* of G' move to G' and then the robots in the *south-region* of G' move to G'. The robots in the *west-region* follow case I and the robots in the *south-region* follow case II to reach and move inside G'. However, as soon as all the robots in the *west-region* reach G', the robots in the *south-region* may not be able to move immediately to G' as there might not be empty positions. Because, in case I, the robots inside G' move South/East to occupy South/East part of G'. But, when there are no robots in the *west-region* of G', the robots inside G' also satisfy case II and start moving North/West. This may take at most $2k$ time to have empty nodes in the southmost horizontal line L'_N of G'. As soon as there are empty nodes on L'_N, the robots in the *south-region* start moving to G' following case II. Case I and II execute for $\leq 2n$ rounds each. Thus, all the robots reach gathering configuration in $\leq 4n+2k$ rounds and Phase 1.2 terminates.

Hence, Phase 1 finishes in total at most $O(n)+4n+2k = O(n)$ rounds. □

Phase 2 (Pre-Scatter). The pseudocode of the algorithm for Phase 2 is given in Algorithm 6. The purpose of Phase 2 is to distribute the robots on $k+1$ vertical lines separated at d distance apart, such that each vertical line contains $k+1$ robots achieving the pre-scatter configuration $C_{pre-scatter}$ (Fig. 3(c)). In this phase, robots move horizontally West in G. Let $H(r_i)$ and $V(r_i)$ be the horizontal and vertical line passing through r_i in G, respectively. Let L_N be the north boundary line of G and L'_S be the horizontal line parallel to L_N and passing through k distance South of L_N. In each LCM cycle, r_i moves one unit West if the node is empty and it sees a robot on $H(r_i)$ in the East at distance less than d. When a robot reaches to the West boundary line L_W, it changes its state to Phase 3. r_i also changes its state to Phase 3, if it sees a robot in the South of L'_S at horizontal distance $d \cdot x$ from $V(r_i)$ where $x = 0, 1, 2, \ldots$. We prove the following lemma.

Algorithm 6: PRE_SCATTER($r_i, d, H(r_i), V(r_i), L_E, L_W, L'_S, C(r_i)$)

if $(x_i, y_i) \in L_W$ **then** $r_i \cdot state \leftarrow 2$;
else if r_i sees a robot in the south of L'_S at distance $d \cdot x$ from $V(r_i)$ (on, left or right of $V(r_i)$) where $x = 0, 1, 2, \ldots$ **then** $r_i \cdot state \leftarrow 2$;
else if $(x_i - 1, y_i)$ is empty $\wedge$ r_i sees a robot on $H(r_i)$ at distance less than d in the East **then** r_i moves to $(x_i - 1, y_i)$;

Algorithm 7: SCATTER($r_i, d, V(r_i), L_N, L_S, C(r_i)$)

$L_V^d \leftarrow$ vertical line parallel to $V(r_i)$ at distance d east of $V(r_i)$;
if $(x_i, y_i) \in L_N \vee (x_i, y_i) \in L_S$ **then** r_i terminates;
else if r_i sees robots on $V(r_i)$ exactly at distance $d \cdot x$ from r_i where $x = 1, 2, 3, \ldots$ **then**
  r_i terminates;
else if r_i sees a robot on $V(r_i)$ in the North at distance less than $d \wedge r_i$ sees no robot between $V(r_i)$ and $L_V^d \wedge (x_i, y_i - 1)$ is empty **then** r_i moves to $(x_i, y_i - 1)$;

Lemma 5. *Phase 2 finishes in $O(n)$ rounds avoiding robot collisions.*

Phase 3 (Scatter). Phase 3 executes after Phase 2 and the pseudocode is given in Algorithm 7. The purpose of Phase 3 is to uniformly scatter the robots in G achieving the Uniform Scattering configuration as depicted in Fig. 3(d). In this phase, robots move vertically towards South in G. Let L_N, L_S be the North and South boundary lines of G, respectively and $H(r_i), V(r_i)$ be the horizontal and vertical lines passing through r_i, respectively. Let L_V^d be the vertical line parallel to $V(r_i)$ and passing through d distance East of $V(r_i)$. All the robots on L_N terminate without moving as they are already at the final positions. When a robot r_i at (x_i, y_i) sees another robot on $V(r_i)$ in the North at distance less than d, it moves one unit South to $(x_i, y_i - 1)$ if the node is empty and r_i sees no robot between $V(r_i)$ and L_V^d. When r_i reaches to L_S, it terminates. r_i also terminates when it sees all the robots on $V(r_i)$ (up to the visibility range) are exactly at d distance apart.

Lemma 6. *Phase 3 finishes in $O(n)$ rounds avoiding robot collisions.*

Proof of Theorem 1. The analysis above proves Theorem 1 for the $\mathcal{FSYNC}$. $\square$

4 Uniform Scattering Algorithm in $\mathcal{ASYNC}$

In this section, we extend the algorithm for the $\mathcal{FSYNC}$ to the $\mathcal{ASYNC}$ setting. We describe a collision-free, time-optimal $\mathcal{ASYNC}$ $O(n)$-epoch algorithm. The algorithm has four phases: Phase 0 (Pre-Gather), Phase 1 (Gather), Phase 2 (Pre-Scatter) and Phase 3 (Scatter). Unlike $\mathcal{FSYNC}$, the $\mathcal{ASYNC}$ algorithm has one more phase called Phase 0 (Pre-Gather) before Phase 1. Phases 1, 2 and 3 of the $\mathcal{ASYNC}$ are equivalent to the $\mathcal{FSYNC}$ but each phase is modified appropriately. In the $\mathcal{FSYNC}$ algorithm, all the robots switch to Phase 2 from Phase 1 synchronously when C_{gather} is achieved. But in the $\mathcal{ASYNC}$ algorithm,

Phase 0: *Pre-Gather*
- All the robots on L_N of G move South to L'_N and reach Phase 1 avoiding collision.

Phase 1: *Gather*
- All the robots perform *Northeast move* (avoiding collisions due to the movement of robots at L_N to L'_N in Phase 0) to reach sub-grid G' in the North-East part of G below L_N.
- When a robot reaches G', it performs *Balancing move* inside G' to achieve C_{gather}.
- All the robots reach Phase 2 after achieving gathering configuration C_{gather}.

Phase 2: *Pre-Scatter*

Phase 2.1:
- Robot at the North-East corner v'_{ne} of G' moves North to the North-East corner v_{ne} of G.
- Robot at the South-West corner v'_{sw} of G' moves West after the robot at v'_{ne} moved to v_{ne}.
- Then, the remaining robots on the westmost boundary line L'_W of G' move West.
- After all the robots on L'_W moved one unit west of L'_W, the southmost robot among them moves further West; the remaining others in the column also follow the West move after it.

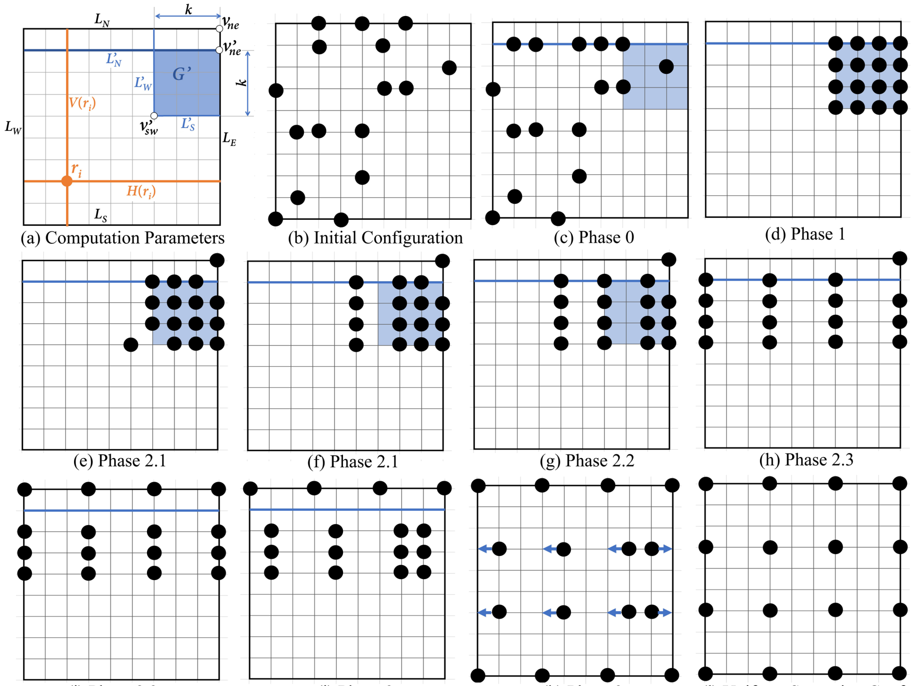
Phase 2.2:
- When a robot inside G' sees next robot in the West on its horizontal line at distance 3 and all the nodes in the East are occupied, it moves one unit West.
- The robot also moves West when it sees the next robot in the East on its horizontal line has already started moving West.

Phase 2.3:
- A robot moves West when it sees another robot in the East on its horizontal line at distance less than d.
- When the $k + 1$ robots reach the westmost boundary line L_W of G, the northmost robot among them moves North to L_N.
- Among every next column of $k + 1$ robots at d distance apart, the northmost robot moves North to L_N after seeing the robot from the previous column at distance d moved to L_N.
- The robots on L_N reach Phase 3. The remaining robots on L_E move one unit West and reach Phase 3. The other remaining robots move one unit East and reach Phase 3.

Phase 3: *Scatter*
- Each robot moves South avoiding collision and maintaining a gap of at most distance d to the next robot in the North on its vertical line.
- When a robot reaches the south boundary line L_S of G, if it lies one unit West of L_E, it moves one unit East and terminates; Otherwise, it moves one unit West and terminates.
- Any other robot when sees no robot in South on its vertical line but a robot at d distance South on the next vertical line in the West (East), it moves one unit West (East) and terminates.

Fig. 5. Algorithm for UNIFORM SCATTERING in the $\mathcal{ASYNC}$ setting.

robots may become active asynchronously and some robots may never see C_{gather} when they become active. So, we need a different mechanism to switch from Phase 1 to Phase 2 in $\mathcal{ASYNC}$. To handle this situation, we introduce Phase 0 before Phase 1 which makes the northmost boundary line of G empty. Later in Phase 1, when C_{gather} is achieved, one robot is moved to the North-East corner of G which becomes a reference for other robots to switch from Phase 1 to Phase 2. The detail mechanism is explained later in the description of each Phase. Each robot passes through Phases 0–3 sequentially. Figure 5 outlines the algorithm in high level. Figure 6 illustrates the configuration of robots at different stages of each phase.



(a) Computation Parameters (b) Initial Configuration (c) Phase 0 (d) Phase 1
(e) Phase 2.1 (f) Phase 2.1 (g) Phase 2.2 (h) Phase 2.3
(i) Phase 2.3 (j) Phase 3 (k) Phase 3 (l) Uniform Scattering Conf.

Fig. 6. Configuration of robots at different phases executing algorithm for $\mathcal{ASYNC}$.

Phase 0 (Pre-Gather). The purpose of this phase is to make the North boundary line L_N of G empty. If any robot r_i is located on L_N in C_{init}, the robot is moved South during Phase 0. For this, first, r_i checks if the South neighboring node on the next horizontal line below L_N (i.e. L'_N) is empty or not. If the South node is empty, r_i moves to it. Otherwise, if the West neighboring node is empty, r_i moves one unit West on L_N. The movement of r_i on L_N towards West helps it to find the empty node on L'_N fast because the robots on L'_N move East. Once a robot moves South of L_N, it never moves again to L_N. Phase 0 ends when no robot is positioned on L_N (e.g. Fig. 6(c)).

Lemma 7. *Phase 0 ends in $O(n)$ epochs avoiding robot collisions.*

Phase 1 (Gather). Similar to the Phase 1 of $\mathcal{FSYNC}$, the purpose of this phase is to reach a gathering configuration C_{gather} on the North-East part of G; the only difference is that the sub-grid G' for C_{gather} in $\mathcal{ASYNC}$ lies one unit South of G' in $\mathcal{FSYNC}$. Let L'_N be the next horizontal line below the North boundary line L_N of G. Then the sub-grid G' is bounded by the East boundary line L_E, L'_N, the vertical line parallel to L_E at k distance West of L_E (say L'_W) and the horizontal line parallel to L'_N at k distance South of L'_N (say L'_S). C_{gather} is said to be achieved if all the robots reach in G' at the distinct nodes. Phase 1 in $\mathcal{ASYNC}$ is also divided into two sub-phases, Phase 1.1 and 1.2 that execute sequentially. We describe each sub-phase below.

Phase 1.1 (Northeast moves). This phase is analogous to the Phase 1.1 of $\mathcal{FSYNC}$ after removing the North boundary line L_N of G. A robot r_i never moves North towards L_N from L'_N. If r_i is already on L_N in C_{init}, it moves South to L'_N during Phase 0. Any robot below L_N (except the robot at L'_N) first searches empty position on its North neighboring node for possible North move. If the North move is not possible, it searches an empty node in the East neighboring node for possible East move. A robot below L'_N does not move North to L'_N if it sees a robot on L_N in the same vertical line. Similarly, a robot at L'_N does not move East if it sees a robot on L_N in its North-East neighboring node. This handles the possible collision due to movement of robot at L_N to L'_N.

Phase 1.2 (Balancing moves). Phase 1.2 of $\mathcal{ASYNC}$ directly follows Phase 1.2 of $\mathcal{FSYNC}$ to reach the gathering configuration C_{gather}. Since robots perform their LCM cycles asynchronously, how they change their states to reach Phase 2 in $\mathcal{ASYNC}$ is slightly different than in $\mathcal{FSYNC}$. If a robot r_i sees C_{gather} configuration, it changes its state to Phase 2. Otherwise, r_i changes its state to Phase 2 after seeing the robot in the North-East corner v'_{ne} of G' moved North to L_N. In the mean time, r_i ensures that there is no robot in the South of G', the remaining k nodes on L_E of G' are occupied and the nodes in the East of r_i on $H(r_i)$ (except v'_{ne}) are also occupied.

Complying with Lemma 2–4, we have following lemma for Phase 1 in $\mathcal{ASYNC}$:

Lemma 8. *Phase 1 finishes in O(n) epochs. Phase 1 is collision-free and deadlock-free.*

Phase 2 (Pre-Scatter). In this phase, robots move West to reach the pre-scatter configuration $C_{pre-scatter}$. Unlike $\mathcal{FSYNC}$, in $C_{pre-scatter}$ of $\mathcal{ASYNC}$, the horizontal line below L_N (i.e. L'_N) is empty, instead the horizontal line at $k+1$ distance South of L_N (i.e. L'_S) contains the $k+1$ robots separated at d distance apart. Figure 6(e–i) illustrate the movements of robots during Phase 2 to reach $C_{pre-scatter}$ from C_{gather}.

Phase 2 is divided into three sub-phases, Phase 2.1–2.3. In Phase 2.1, only the robots on L'_W and the robot at the North-East corner of G' (say v'_{ne}) are moved. The robot at v'_{ne} moves North to the North-East corner of G (say v_{ne}) and reaches Phase 2.3. Then, the robot at the South-West corner of G' (say v'_{sw}) moves one unit West to the neighboring node (say v_{ref}). After that, the remaining robots on L'_W also move one unit West. Now, the robot at v_{ref} sees all the nodes on its vertical line towards North occupied and L'_W empty, then it moves one unit West and reaches Phase 2.2. The remaining robots in the North of v_{ref} also move one unit West after it and reach Phase 2.2.

In Phase 2.2, when a robot r_i is inside G' and sees next robot in the West on the same horizontal line $H(r_i)$ at distance 3, it moves one unit West and waits for next robot in the East on $H(r_i)$ to move one unit West. When r_i sees the next robot in the East on $H(r_i)$ moved one unit West (i.e. r_i sees a robot at distance 2 in the East on $H(r_i)$), it also moves one unit West and reaches Phase 2.3.

In Phase 2.3, robot r_i moves West when it sees another robot in the East at distance less than d on $H(r_i)$. Since v'_{ne} is empty, as a special case, the eastmost robot on L'_N can move up to d distance West of L_E without seeing a robot in the East on L'_N. When the westmost $k+1$ robots reach L_W of G, the northmost robot among them moves North to L_N. Among every next column of $k+1$ robots at d distance apart, the northmost robot moves to L_N. Then, the pre-scatter configuration ($C_{pre-scatter}$) is achieved. Since, in every 2 epochs, at least one column of robots move one unit West, it is immediate that the westmost $k+1$ robots reach L_W in $2(n-k)$ epochs. In next $2k$ epochs, the k robots on L'_N move to L_N. Thus, $C_{pre-scatter}$ is achieved in at most $2n$ epochs.

All the robots change their states to Phase 3 after achieving $C_{pre-scatter}$. Additionally, if $d > 2$, any robot r_i south of L_N moves one unit East as well (except the robots on L_E which move one unit West) (Fig. 6(j)). Thus, Phase 2 also finishes in $O(n)$ epochs in $\mathcal{ASYNC}$. Since no robot moves South in Phase 2 and no robot reaches Phase 3 before $C_{pre-scatter}$, the movements of robots in Phase 2 of $\mathcal{ASYNC}$ are collision-free.

Lemma 9. *Phase 2 finishes in $O(n)$ epochs in $\mathcal{ASYNC}$ avoiding robot collisions.*

Phase 3 (Scatter). In this phase, robots move South to achieve UNIFORM SCATTERING configuration and terminate. Figure 6(j–l) provide an illustration. If $d = 2$, robot r_i has the visibility range of n and hence can see all the final positions on it's vertical line. Then, r_i moves South by directly following the Phase 3 of $\mathcal{FSYNC}$ to reach the final position and terminates. If $d > 2$, the algorithm works as follow: Let $H(r_i), V(r_i)$ be the horizontal and vertical lines passing through r_i, respectively. Let $V'(r_i)$ be the vertical line parallel to $V(r_i)$ and passing through one unit West of r_i (for the robots at one unit West of L_E, consider L_E as $V'(r_i)$). When a robot r_i at (x_i, y_i) sees another robot on $V(r_i)$ in North at distance less than d, then r_i moves to $(x_i, y_i - 1)$ if $(x_i, y_i - 1)$ and $(x_i - 1, y_i - 1)$ are both empty. If r_i it on one unit West of L_E, it ensures that $(x_i + 1, y_i - 1)$ is empty instead of $(x_i - 1, y_i - 1)$ to move South. When r_i reaches the south boundary line L_S, it moves West (except the eastmost robot on L_S which moves East to L_E) to reach the final position and terminates. When r_i sees another robot r_j on $V'(r_i)$ at exactly d distance South of $H(r_i)$, r_i moves horizontally to $V'(r_i)$ to occupy the final position and terminates. The southmost robots on the $k+1$ vertical lines take at most $2(n-k)$ epochs to reach L_S. By that time all the robots on each of those $k+1$ vertical lines are at d distance apart. In at most next $2k$ epochs, all those robots reach to the final positions and terminate. Thus, in at most $2n$ epochs, Phase 3 terminates.

Lemma 10. *Phase 3 finishes in $O(n)$ epochs in the $\mathcal{ASYNC}$ setting.*

Proof of Theorem 1. Combine results of Lemmas 7, 8, 9 and 10. □

5 Concluding Remarks

We have provided the first optimal $O(n)$ time algorithm to the UNIFORM SCATTERING problem in a square grid graph of $N = (n+1) \times (n+1)$ nodes in the COR model under the $\mathcal{ASYNC}$ setting. This is the $O(n/d) = O(k)$ improvement compared to the best previously known algorithm with runtime $O(N/d) \equiv O(n^2/d)$ in the COR model. In the future work, it will be interesting to extend our algorithm to consider faults.

References

1. Barrameda, E.M., Das, S., Santoro, N.: Uniform dispersal of asynchronous finite-state mobile robots in presence of holes. In: ALGOSENSORS, pp. 228–243 (2013)
2. Barriere, L., Flocchini, P., Mesa-Barrameda, E., Santoro, N.: Uniform scattering of autonomous mobile robots in a grid. In: IPDPS, pp. 1–8 (2009)
3. Cohen, R., Peleg, D.: Local spreading algorithms for autonomous robot systems. Theor. Comput. Sci. **399**(1–2), 71–82 (2008)
4. Das, S., Flocchini, P., Prencipe, G., Santoro, N., Yamashita, M.: Autonomous mobile robots with lights. Theor. Comput. Sci. **609**, 171–184 (2016)
5. Défago, X., Souissi, S.: Non-uniform circle formation algorithm for oblivious mobile robots with convergence toward uniformity. Theor. Comput. Sci. **396**(1–3), 97–112 (2008)
6. Elor, Y., Bruckstein, A.M.: Uniform multi-agent deployment on a ring. Theor. Comput. Sci. **412**(8–10), 783–795 (2011)
7. Flocchini, P., Prencipe, G., Santoro, N.: Self-deployment of mobile sensors on a ring. Theor. Comput. Sci. **402**(1), 67–80 (2008)
8. Flocchini, P., Prencipe, G., Santoro, N.: Distributed Computing by Oblivious Mobile Robots. Synthesis Lectures on Distributed Computing Theory, vol. 3, no. 2, pp. 1–185 (2012)
9. Flocchini, P., Prencipe, G., Santoro, N., Viglietta, G.: Distributed computing by mobile robots: uniform circle formation. Distrib. Comput. **30**(6), 413–457 (2016). https://doi.org/10.1007/s00446-016-0291-x
10. Flocchini, P., Prencipe, G., Santoro, N., Widmayer, P.: Arbitrary pattern formation by asynchronous, anonymous, oblivious robots. Theor. Comput. Sci. **407**(1–3), 412–447 (2008)
11. Heo, N., Varshney, P.K.: Energy-efficient deployment of intelligent mobile sensor networks. Trans. Sys. Man Cyber. Part A **35**(1), 78–92 (2005)
12. Howard, A., Matarić, M.J., Sukhatme, G.S.: An incremental self-deployment algorithm for mobile sensor networks. Auton. Rob. **13**(2), 113–126 (2002)
13. Hsiang, T.-R., Arkin, E.M., Bender, M.A., Fekete, S.P., Mitchell, J.S.B.: Algorithms for rapidly dispersing robot swarms in unknown environments. In: Boissonnat, J.-D., Burdick, J., Goldberg, K., Hutchinson, S. (eds.) Algorithmic Foundations of Robotics V. STAR, vol. 7, pp. 77–93. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-45058-0_6
14. Izumi, T., Kaino, D., Potop-Butucaru, M.G., Tixeuil, S.: On time complexity for connectivity-preserving scattering of mobile robots. Theor. Comput. Sci. **738**, 42–52 (2018)

15. Katreniak, B.: Biangular circle formation by asynchronous mobile robots. In: Pelc, A., Raynal, M. (eds.) SIROCCO 2005. LNCS, vol. 3499, pp. 185–199. Springer, Heidelberg (2005). https://doi.org/10.1007/11429647_16
16. Lin, Z., Zhang, S., Yan, G.: An incremental deployment algorithm for wireless sensor networks using one or multiple autonomous agents. Ad Hoc Netw. **11**(1), 355–367 (2013)
17. Poduri, S., Sukhatme, G.S.: Constrained coverage for mobile sensor networks. In: ICRA, pp. 165–171 (2004)
18. Poudel, P., Sharma, G.: Time-optimal uniform scattering in a grid. In: ICDCN, pp. 228–237 (2019)
19. Sharma, G., Krishnan, H.: Tight bounds on localized sensor self-deployment for focused coverage. In: ICCCN, pp. 1–7. IEEE (2015)
20. Shibata, M., Mega, T., Ooshita, F., Kakugawa, H., Masuzawa, T.: Uniform deployment of mobile agents in asynchronous rings. In: PODC, pp. 415–424 (2016)
21. Sinan Hanay, Y., Gazi, V.: Distributed sensor deployment using potential fields. Ad Hoc Netw. **67**(C), 77–86 (2017)
22. Suzuki, I., Yamashita, M.: Distributed anonymous mobile robots: formation of geometric patterns. SIAM J. Comput. **28**(4), 1347–1363 (1999)