Coalescing Walks on Rotor-Router Systems*

Colin Cooper[1], Tomasz Radzik[1], Nicolás Rivera[1], and Takeharu Shiraga[2]

[1] Department of Informatics, King's College London, United Kingdom
{colin.cooper,tomasz.radzik,nicolas.rivera}@kcl.ac.uk
[2] Theoretical Computer Science Group, Department of Informatics,
Kyushu University, Fukuoka, Japan
shiraga@tcslab.csce.kyushu-u.ac.jp

Abstract. We consider the rotor-router mechanism for distributing particles in an undirected graph. If the last particle passing through a vertex v took an edge (v, u), then the next time a particle is at v, it will leave v along the next edge (v, w) according to a fixed cyclic order of edges adjacent to v. The system works in synchronized steps and when two or more particles meet at the same vertex, they coalesce into one particle. A k-particle configuration of such a system is *stable*, if it does not lead to any coalescing. For $2 \leq k \leq n$, we give the full characterization of stable k-particle configurations for cycles. We also show sufficient conditions for regular graphs with n vertices to admit n-particle stable configurations.

1 Introduction

We consider an undirected connected graph $G = (V, E)$ and the *rotor-router* mechanism which keeps moving simple entities along the edges of G in synchronized steps. We call these entities *particles*, but terms like agents, tokens or chips may be used by others. Each edge $\{v, u\}$ is viewed as a pair of opposite arcs (v, u) and (u, v), and for each vertex, the arcs outgoing from this vertex are kept in a fixed cyclic order. In each step, each particle moves from its current vertex to an adjacent vertex. For each vertex v, if (v, u) is the most recently traversed arc outgoing from v, then the next particle leaving v will traverse the next arc (v, w). This is implemented by maintaining at each vertex v the *vertex pointer* π_v which indicates which arc outgoing from v should be taken next. While particles keep passing through v, the pointer π_v is the "rotor" moving around the cyclic order of arcs outgoing from v. This model was introduced by Priezzhev *et al.* [17], was further studied and popularised by James Propp, and hence also referred to as the *Propp machine*.

The rotor-router mechanism can be viewed as a model of graph exploration by simple mobile entities and the efficiency of such exploration has been extensively studied. While the earlier works refer mostly to single-particle rotor-router exploration, there are now also a few recent results concerning the multi-particle

* This work was supported in part by EPSRC grant EP/M005038/1, "Randomized algorithms for computer networks".

C. Scheideler (Ed.): SIROCCO 2015, LNCS 9439, pp. 444–458, 2015.
DOI: 10.1007/978-3-319-25258-2_31

case. In both the single-particle and the multi-particle cases, some similarities with graph exploration by random walks have been observed. This further motivates investigations of the rotor-router mechanism as a possible deterministic alternative to random walks. For example, one random walk on a cycle of length n covers (visits) all vertices in expected $\Theta(n^2)$ time and a single-particle rotor-router does this in deterministic $\Theta(n^2)$ worst-case time. Wagner *et al.* [18, 19] showed that for an arbitrary connected n-vertex m-edge graph and an arbitrary initial configuration of the single-particle rotor-router system (arbitrary cyclic orders of arcs, arbitrary initial setting of the vertex pointers and an arbitrary starting vertex for the particle) the particle visits all vertices of this graph in $O(nm)$ steps. Subsequently a number of more detailed analyses of single-particle rotor-router systems in various types of graphs have been published [4–6,20], but it can be shown that $\Theta(nm)$ is the worst-case bound for the general graphs. The expected cover time by a random walk has the same general bound $O(nm)$. More recently the cover time by (parallel) k random walks has been analyzed and speedups over a single random walk between $\Theta(\log k)$ and $\Theta(k)$ have been shown for various classes of graphs and various initial settings [3, 9, 11, 12]. The similar range of speedups for k-particle rotor router (over the single-particle system) have been demonstrated in [10, 14].

In this paper we look at another aspect of multi-particle systems: *coalescence of particles*. Whenever two or more particles meet at the same step in the same vertex of the graph, then they coalesce (merge) into one particle. This particle continues moving through the graph, following the underlying protocol (for example, the random-walk protocol or the rotor-router mechanism). Coalescing random walks is a long established topic, attracting research interest partly due to its close relation to the randomized *pull-voting* process [1]. The main case considered is when initially each vertex has one particle and the question is to provide good bounds on the expected time (number of steps) until full coalescing into one particle. Aldous [2] conjectured that this expected full-coalescence time is at most of the order of the maximum hitting time of a single random walk. This conjecture has not been fully settled yet, but considerable progress has been made [8, 16].

Systems with coalescing particles may find applications in parallel computing. For example, Israeli and Jalfon [13] proposed coalescing random walks as the basis of a self-stabilizing mutual exclusion algorithm. In a network of interconnected processing units (PUs) competing for access to some resource, each PU creates a token and sends it for a random walk through the network. Tokens coalesce whenever they meet at the same PU, eventually only one token remains in the network (provided the network is connected and non-bipartite) and the PU with the token gets exclusive access to some resource.

In this paper we want to initiate investigation of rotor-router systems with coalescing particles. While the coalescing random walks will always eventually merge into a single walk (or two walks in the case of bipartite graphs), and good bounds for expected coalescence time are known, it is not difficult to come up with examples of rotor-router configurations with multiple particles which do

not lead to any coalescence. Thus a reasonable first question is to characterize such stable multiple-particle configurations. Other interesting questions are to bound the probability that a random initial configuration (defined, for example, by random initial settings of the vertex pointers) leads to full coalescing, and to analyze the coalescence time. We give some answers to the first question, leaving the other two as directions for further research. In particular, we give a full characterization of stable configurations in cycles. This characterization implies that if the length n of the cycle is prime, then any initial configuration with $k < n$ particles leads to full coalescing. We also show that all n-vertex even-degree regular graphs admit n-particle stable configurations and give a sufficient condition for odd-degree regular graphs to admit such configurations. For graphs which have vertices of degree greater than 2 and for $k < n$ particles, the full coalescence may depend more on the structure of the graph than on the primality of the number of edges or the number of vertices. As an example of this, we show a graph with m edges and constant maximum degree, which admits k-particle stable configuration, for any sufficiently large m and any $2 \leq k \leq \sqrt[3]{m/6}$.

The important property of non-coalescing rotor-router systems which we use in our work is the long run behaviour of such systems. For single-particle rotor-router systems, Bhatt *et al.* [6] showed that within $O(nm)$ steps, the particle *enters (establishes) an Eulerian cycle*. More precisely, after the initial *stabilisation period* of $O(nm)$ steps, the particle keeps repeating the same Eulerian cycle of the whole set $\overrightarrow{E}$ of directed arcs. The long run behaviour of multiple-particle rotor-routers was open for a long time, but has been recently settled by Chalopin et al. [7]. They showed that in polynomial number of steps the system reaches a stable configuration S, that is a configuration which will be repeated after some (potentially exponential) number of steps. Most importantly, they provide a strong characterization of the way the particles will be moving around the graph starting from a stable configuration. The set $\overrightarrow{E}$ can be partitioned into arc-disjoint Eulerian circuits and the particles can be assigned to these circuits such that each particle will be perpetually following the circuit it is assigned to. The circuits are arc disjoint, but may share vertices, and two or more particles can be assigned to the same circuit. Our analysis of coalescing rotor-router systems is directly based this characterization of the stable configurations of non-coalescing rotor-router systems.

2 Preliminaries

We consider an undirected, simple (no loops or multiple edges), connected graph $G = (V, E)$ with $n \geq 3$ vertices and m edges. We define $\overrightarrow{E} = \{(v, w), (w, v) : \{v, w\} \in E\}$ as the set of (directed) arcs in G. For each $v \in V$, the arcs outgoing from v are arranged in a fixed cyclic order. The vertex pointer π_v indicates the arc outgoing from v which will be taken by the next particle leaving v. When a particle leaves v along the arc indicated by the pointer π_v, the pointer advances to the next arc outgoing from v. The system works in synchronised steps and each particle moves in each step (that is, a particle never waits in the same

vertex). In a coalescing rotor-router system, if two or more particles arrive at the same vertex v at the same step, they coalesce into one particle, which moves out of v in the next step in the direction indicated by the vertex pointer. In a non-coalescing system, if $q \geq 2$ particles arrive at the same vertex v at the same time, they all leave v in the next step taking the q consecutive arcs outgoing from v, starting from the arc indicated by the vertex pointer and wrapping-around, if q is greater than the degree of v. The vertex pointer changes to the arc next after the last arc taken by a particle. We do not distinguish among particles, so the order in which the particles leave vertex v in the same step is not important.

A k-particle *configuration* S is defined by the values of the vertex pointers and the position of the particles. For a configuration S, we denote by $\sigma(S)$ the set of all configurations visited starting from S (we assume the system works perpetually). We say that a configuration S is *stable*, if after starting from S we eventually return to S. Clearly, a configuration S is stable, if and only if, for each $S' \in \sigma(S)$, $\sigma(S') = \sigma(S)$. A set of configurations is stable, if it is equal to $\sigma(S)$ for a stable configuration S. Two configurations of a rotor-router system are isomorphic, if there is a one-to-one mapping on V which preserves the cyclic orders of arcs, the vertex pointers and the particle counts at vertices. We say that a graph G admits a k-particle stable configuration, if there exist cyclic orders of arcs at the vertices of G, the initial positions of vertex pointers and the initial locations of k particles, which define a k-particle stable configuration of the coalescing rotor-router system.

By definition, the rotor-router system is *locally fair*, sending, in the long run, the same number of particles into each of the arcs outgoing from the same vertex. More precisely, if S is a stable configuration and it takes T steps to return to S, then during these T steps each of the arcs outgoing from the same vertex is traversed the same number of times (otherwise the vertex pointer does not return to its initial position). This local fairness implies the *global fairness*.

Lemma 1. *If S is a stable configuration and it takes T steps to return to S, then during these T steps each arc has been traversed the same number of times.*

Proof. For each vertex v, during these T steps each arc outgoing from v has been traversed the same number of times. Let $\alpha(v)$ denote this number, let $\alpha_{\min} = \min\{\alpha(v) : v \in V\}$ and assume, by contradiction, that $U = \{v \in V : \alpha(v) = \alpha_{\min}\} \neq V$. Each arc from U to $V \setminus U$ has been traversed $\alpha_{\min}$ times but each arc from $V \setminus U$ to U has been traversed more than $\alpha_{\min}$ times. This contradicts the assumption that after T steps we are back in the initial configuration S, so the number of traversals from U to $V \setminus U$ must have been the same as the number of traversals from $V \setminus U$ to U.

Chalopin et al. [7] proved the following strong characterization of stable configurations of non-coalescing rotor-router systems, which is valid also for the coalescing systems.

Theorem 1. [7] *A configuration S is stable, if and only if, there exists a decomposition of $\overrightarrow{E}$ into arc-disjoint Eulerian circuits and an assignment of particles*

to circuits (possibly with more than one particle assigned to the same circuit) such that starting from S, each particle follows perpetually the circuit to which it is assigned.

This theorem says that while a rotor-router system keeps changing from configuration to configuration within a stable set, each particle keeps tracing the same Eulerian circuit. The circuits are arc disjoint and cover the whole set of arcs $\overrightarrow{E}$. Two or more particles can trace the same circuit and each circuit must be traced by at least one particle (see Lemma 1). Note that opposite arcs (v, w) and (w, v) may belong to the same circuit or to two different circuits.

Corollary 1. *Let $C_0, C_1, \ldots, C_{q-1}$ be a circuit decomposition associated with a stable set σ and let $k_i \geq 1$ denote the number of particles which follow the circuit C_i. Then the ratio $|C_i|/k_i$ is the same for each circuit.*

For a stable configuration of a coalescing rotor-router system, the Eulerian circuit decomposition of Theorem 1 must be unique. Note also that while each stable configuration has an associated Eulerian circuit decomposition, the converse is not true. A decomposition of $\overrightarrow{E}$ into arc-disjoint Eulerian circuits might not correspond to any stable configuration, because it might not be possible to set up the vertex pointers and the initial positions of particles to make the particles follow the circuits.

3 Stable Configurations in a Cycle

We consider the coalescing rotor-router system based on the n-vertex cycle $\mathcal{C}_n$. We first show various types of stable k-particle configurations and then prove that these are the only possible stable configurations. Throughout this section we assume that $n \geq 3$ and $k \geq 2$. Let $\mathcal{C}_n = (v_0, \ldots, v_{n-1}, v_0)$, so $\overrightarrow{E} = \{(v_i, v_{i+1}), (v_{i+1}, v_i) : i = 0, 1, \ldots, n-1\}$, assuming $v_n \equiv v_0$. Each decompositions of $\overrightarrow{E}$ into arc-disjoint Eulerian circuits is either of the *Cycle type* (the C type) or the *Path type* (the P type), with the latter split further into two categories P1 and P2.

C: Two Eulerian circuits $(v_0, \ldots, v_{n-1}, v_0)$ and $(v_0, v_{n-1}, v_{n-2}, \ldots, v_0)$.

P: The Eulerian circuits are defined by a partitioning of the edges of the cycle $\mathcal{C}_n$ into edge-disjoint paths $P_0, ..., P_{q-1}$, $q \geq 1$. The last vertex of path P_i is the first vertex of path P_{i+1}, for $i = 0, 1, \ldots, q-1$, with $P_q \equiv P_0$. Each of these paths $P = (w_0, w_1, ..., w_j)$ defines the Eulerian circuit $(w_0, w_1, \cdots, w_{j-1}, w_j, w_{j-1}, \cdots, w_1, w_0)$.

P1: There is only one path, which covers the whole cycle, that is, there is only one Eulerian circuit, which covers all arcs in $\overrightarrow{E}$. Each such circuit is isomorphic to the circuit $(v_0, v_1, \cdots, v_{n-1}, v_0, v_{n-1}, \cdots, v_1, v_0)$.

P2: There are at least two paths, so there are at least two circuits in the decomposition.

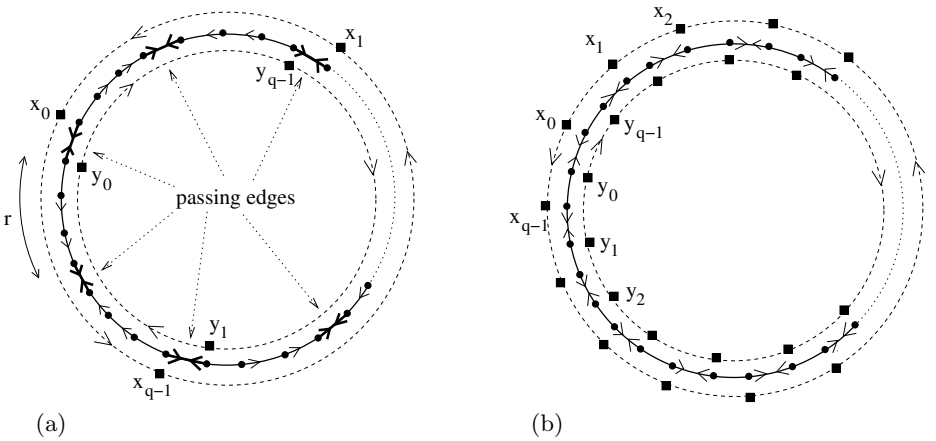

Fig. 1. Stable k-particle configurations of type C, with $k = 2q$ and $n = k(r + 1)$, for integers $q \geq 1$, $r \geq 0$. The particles on one circuit are passing the particles on the other circuit when traversing the "passing edges." There are r edges between two consecutive passing edges. (a) The case $r \geq 1$. (b) The case $r = 0$: each edge is a passing edge.

We say that a stable configuration, or a stable set, is of type X, if the associated circuit decomposition is of type X. Figure 1 shows configurations representing stable sets of type C. There are $k = 2q$ particles, for an integer $q \geq 1$, which are marked on the diagrams with small black squares. Particles $x_0, x_1, \ldots, x_{q-1}$ are assigned to the anti-clockwise circuit and the particles $y_0, y_1, \ldots, y_{q-1}$ are assigned to the clockwise circuit. The particles are evenly spaced along both circuits, with $2(r+1)$ edges between each two consecutive particles on one circuit, for an integer $r \geq 0$ (in Figure 1(a), $r = 3$). Thus the length of the cycle is $n = k(r + 1)$. Consider the relative positions of particles x and y as shown in Figure 1, when the particles x_0 and y_0 are about to traverse the same edge in opposite directions. The arrow at a vertex shows the direction where the next particle will leave this vertex. In this configuration, each pair of particles x_i and y_{q-i}, for $i = 0, 1, \ldots, q - 1$ (with $y_q \equiv y_0$), is about to traverse the same edge in opposite directions. Such traversing of the same edge in opposite directions is repeated every $r + 1$ steps, and each edge which is traversed at some step by a particle x_i in one direction and a particle y_j in the other direction is called a *passing edge*. There are $n/(r + 1) = k$ passing edges, evenly spaced along the cycle. In the case $r = 0$ (shown in Figure 1(b)), there are n particles in total, one on each vertex, and each edge is a passing edge.

We now show two different stable sets of type P2, one with one particle assigned to each circuit and one with two particles assigned to each circuit. We refer to these two types of stable sets as types P2.1 and P2.2, respectively. Stable sets of type P2.1 are illustrated in Figure 2. The cycle has $n = kr$ vertices, for an integer $r \geq 1$, the circuit decomposition is defined by k paths of equal length r, and each circuit has one particle assigned to it. Consider one configuration, which is defined by the positions of the particles in their circuits. To

avoid collisions, these positions are restricted by the following condition. Let C_i and C_{i+1} be adjacent circuits which share vertex u_{i+1}, and let x_i and x_{i+1} be the particles assigned to these circuits. The distance from x_i to u_{i+1} along the circuit C_i must be different than the distance from x_{i+1} to u_{i+1} along the circuit C_{i+1}. Figure 2 shows two stable configurations of type P2.1, which belong to two different (non-isomorphic) stable sets.

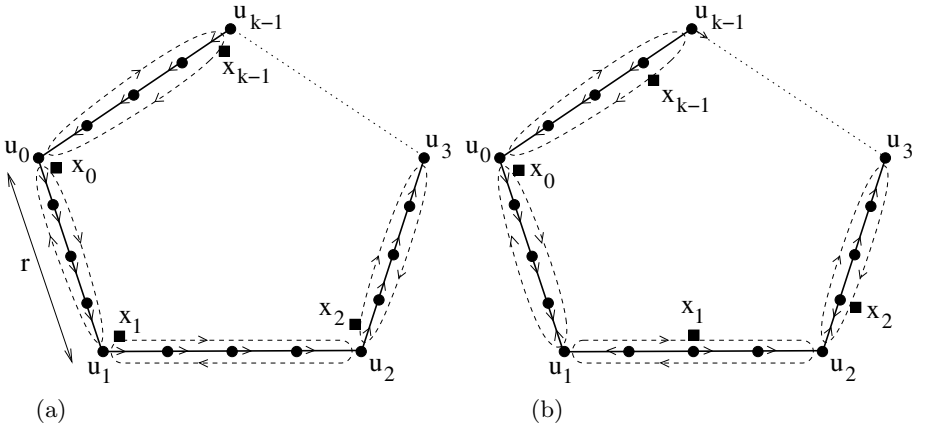

Fig. 2. Stable k-particle configurations of type P2.1; $n = kr$ and $r \geq 1$. Configurations (a) and (b) belong to two different (non-isomorphic) stable sets. In (a), each particle x_i is in the same position within its Eulerian circuits, that is, within the same distance from vertex u_i, moving in the same direction along the cycle (all clock-wise or all anti-clockwise). In (b), the particles are in different positions within their Eulerian circuits.

Figure 3 illustrates stable sets of type P2.2: the Path type, at least 2 circuits and exactly 2 particles in each circuit. The cycle has $n = q(2r + 1)$ vertices, for integers $q \geq 2$ and $r \geq 1$, and the circuit decomposition is defined by q paths of equal length $2r + 1$. Each circuit has two particles assigned to it, so $k = 2q \geq 4$. The two particles x_i and y_i assigned to the same circuit C_i are exactly half-way around the circuit from each other. They will pass each other every $2r + 1$ steps, traversing in opposite directions the middle edge of the path which defines this circuit. To avoid collisions, we have a condition restricting the relative positions of particles on the adjacent circuits, which is analogous as in the stable configurations of type P2.1 described above.

The following theorem, proven in Sections 3.1 and 3.2, gives a full characterization of the stable sets in a cycle.

Theorem 2. *Assume $n \geq 3$ and $2 \leq k \leq n$, and consider the coalescing rotor-router system based on the cycle $\mathcal{C}_n$.*

(i) If k is odd and n is a multiple of k, then there exist k-particle stable sets and they all are of type P2.1 shown in Figure 2.

(ii) If k is even and n is a multiple of k, then there exist k-particle stable sets and each stable set is either of type C shown in Figure 1, or of type P2.1 shown in Figure 2.

(iii) If $k \geq 4$ is even and n is an odd multiple of $k/2$, then there exist k-particle stable sets and they all are of type P2.2 shown in Figure 3.

(iv) For any other combination of n and k, each k-particle configuration leads to at least one coalescing.

Corollary 2. *Consider the coalescing rotor-router system based on the cycle $\mathcal{C}_n$, where $n \geq 3$ is prime. If $2 \leq k \leq n-1$, then each k-particle configuration leads to full coalescing (into one particle). For $k = n$, there is only one unique (up to isomorphism) stable configuration, which is shown in Figure 2(a) with $r = 1$.*

Proof. The first part follows by repeatedly applying the case (*iv*) of Theorem 2, while $k > 1$. The second part follows from the case (*i*) of Theorem 2.

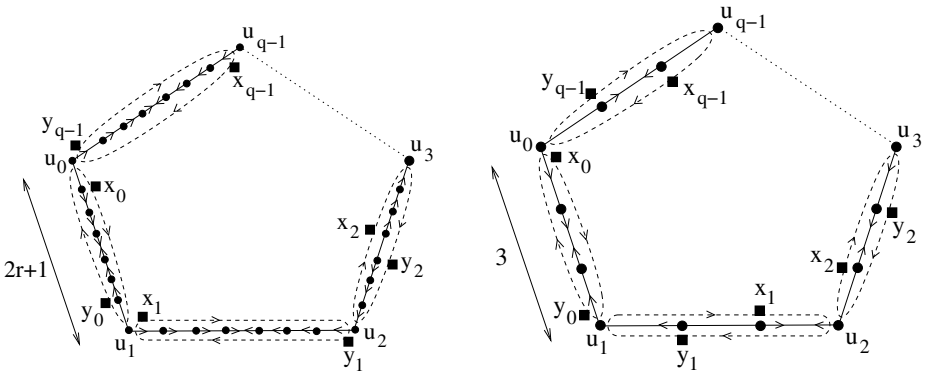

Fig. 3. Stable k-particle configurations of type P2.2, with $k = 2q$ and $n = q(2r+1)$, for integers $q \geq 2$ and $r \geq 1$. (a) The general case $r \geq 1$. (b) The case $r = 1$, where each circuit is in one of three states, illustrated by the three circuits with particles x_0 and y_0, x_1 and y_1, and x_2 and y_2.

3.1 Stable Configurations of the Cycle Type

Lemma 2. *If n is odd or k is odd, then there is no stable set with Eulerian decomposition of type C.*

Proof. Let σ be a k-particle stable set for $\mathcal{C}_n$ with an Eulerian decomposition of type C. Corollary 1 implies that the same number of particles must be assigned to each of the two circuits, so k must be even. Let C_1 and C_2 denote the two circuits and let x be a particle assigned to C_1 and y a particle assigned to C_2. Consider a configuration $S \in \sigma$ such that x and y face each other along an edge $\{v, u\}$: x is at v and will move to u in the next step, while y is at u and will move to v in the next step. If n were odd, then after $(n+1)/2$ steps particles x and y would collide on the opposite side of the cycle.

Lemma 3. *For n and k both even, if there is a stable set with Eulerian decomposition of type C, then each of the two circuits has the same number of particles assigned to it, and the particles assigned to the same circuit must be evenly spaced along this circuit.*

Proof. The condition that each of the two circuits has the same number of particles assigned to it follows from Lemma 1. To prove the second part of the lemma, assume by contradiction that particles on one of the circuits (or on both of them) are not evenly spaced. This implies that there are two consecutive particles x_1 and x_2 on C_1 (x_1 next after x_2 in the direction of C_1) and two consecutive particles y_1 and y_2 on C_2 (y_1 next after y_2 in the direction of C_2) such that x_1 is ahead of x_2 by l_1 arcs and y_1 is ahead of y_2 by l_2 arcs, for some $l_1 \neq l_2$. Assume by symmetry that $l_1 < l_2$ and consider the step when particles x_1 and y_1 have just passed each other, as shown in Figure 4. Particle x_1 is at a vertex v and will be moving towards particle y_2, which is at distance $l_2 - 1$ from x_1. Particle y_1 is at the vertex u next after to vertex v in the direction of circuit C_2, and will be moving towards particle x_2, which is at distance $l_1 - 1$ from y_1. If $l_2 = l_1 + 1$, then either $l_2 - 1$ or $l_1 - 1$ is even, so either particles x_1 and y_2 or particles y_1 and x_2 collide: contradiction. If $l_2 \geq l_1 + 2$, then particle x_2 reaches vertex v before particle y_2 gets there, so x_2 turns back at v, switching from circuit C_1 to C_2: contradiction.

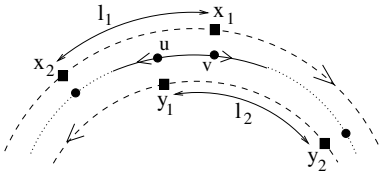

Fig. 4. For the proof of Lemma 3: Eulerian decomposition of type C and particles are not evenly spaced along the circuits.

Lemma 4. *For n and k both even and $k \nmid n$, there is no stable set with Eulerian decomposition of type C.*

Proof. Let n and k be both even and $k \nmid n$, that is, $k = 2q$ and $n = pk + 2r$, for some positive integers q, p, r such that $r < q$. Observe that we must have $q \geq 2$. Assume, by contradiction, that there is a stable set σ with an Eulerian decomposition $\{C_1, C_2\}$ of type C, and start in a configuration $S \in \sigma$. Each of the two circuits has q particles assigned to it and they are evenly spaced along the circuit (Lemma 3). This implies that $q \mid n$, so we must have $q = 2r$. The distance between two consecutive particles on the same circuit is equal to $n/q = 2p + 1$, so it is odd. Consider a particle x on circuit C_1 and two consecutive particles y_1 and y_2 on the other circuit. Since the distance between y_1 and y_2 is odd, then either the distance along circuit C_1 from x to y_1 is even or the distance along C_1 from x to y_2 is even. Thus x will collide with y_1 or y_2: contradiction.

Lemma 5. *For n and k both even and $k \mid n$, there exists a unique (up to isomorphism) stable set with an Eulerian decomposition of type C. This unique stable set is illustrated in Figure 1.*

Proof. Let $k = 2q$ and $n = k(r+1)$ for an integer $r \geq 0$. From Lemma 3, a stable set with an Eulerian decomposition of type C has q particles assigned to each circuit and the distance between any two consecutive particles on the same circuit is equal to $2(r+1)$. This is the stable set illustrated in Figure 1.

3.2 Stable Configurations of the P Type

Lemma 6. *Let σ be a stable set of type P and let C be one of the circuits in the Eulerian decomposition associated with σ. Then no more than two particles are assigned to C.*

Proof. Recall that C spans a path P in the cycle and the particles assigned to C keep walking along the path from one end to another. For any two particles x and y assigned to C, the parity of the distance along P between these particles remains constant. This distance must be odd, because if it were even, then particles x and y would eventually meet. If there were three particles assigned to C, then not all three pairwise distances between these particles could be odd, so two of the three particle would have to meet.

Lemma 7. *Let σ be a stable set of type P and let C be one of the circuits of the Eulerian decomposition associated with σ, and assume that two particles are assigned to C. Then (i) the two particles are at the same time at the opposite ends of the path P which is spanned by C, (ii) path P has an odd number of edges and (iii) the two particles on C always pass each other when traversing (in the opposite directions) the middle edge of the path.*

Proof. Let x and y be the particles assigned to C and let u and v be the end vertices of the path P. To show (i), assume by contradiction that at some step particle x is at vertex v but particle y is not at the other end u. Assume that y is moving towards vertex v. (If y is moving towards u, wait until y reaches u to get an analogous arrangement: y at u and x not at the other end of the path and moving towards u.) Particles x and y will now be moving towards each other, eventually overpassing at step t along some edge $\{w, r\}$: particle x traverses this edge from r to w while particle y traverses from w to r. When now x leaves vertex w to go towards u, the pointer at w is changed to arc (w, r). The distance between w and u is at least the distance between w and v, so the next time a particle comes to w, it will be particle y and it will go back from w to r, contradicting the movement of both particles along the circuit C.

We have shown that at some step particles x and y are the opposite ends of path P. If P had an even $2q$ number of edges, then the particles would meet after the next q steps. The particles must be passing each other when traversing (in the opposite directions) the middle edge of the path, or otherwise they would not be at the end vertices of the path at the same time.

Corollary 3. *There is no stable set of type P1, that is, for each stable set of type P, its Eulerian circuit decomposition must have at least two circuits.*

Lemma 8. *Suppose σ is a stable set of type P2. Then all circuits of the Eulerian decomposition associated with σ have the same length and the same number of particles. That is, each stable set of type P2 is either of type P2.1 shown in Figure 2, or of type P2.2 shown in Figure 3.*

Proof. Assume that the Eulerian decomposition contains a circuit C_1 with one particle and a circuit C_2 with two particles. Then Corollary 1 implies that $|C_1| = |C_2|/2$, but $|C_1|$ is even while Lemma 7 implies that $|C_2|/2$ is odd; contradiction. Thus all circuits in the Eulerian decompositions must have the same number of particles, either one or two, and Corollary 1 further implies that they all must have the same length.

Lemma 9.

(i) If n is a multiple of k, then each stable set of type P2 is of type P2.1.
(ii) If k is even and n is an odd multiple of $k/2$, then each stable set of type the P2 is of type P2.2.

Proof. Lemma 8 says that each stable set of type P2 must be either P2.1 or P2.2. Part *(i)* holds because if we have a stable set of type P2.2, then k is even and Lemma 7 implies that $n = (k/2)(2r+1)$, for some integer $r \geq 0$, so n is not a multiple of k. Part *(ii)* holds because if we have a stable set of type P2.1, then Lemma 8 implies that n is a multiple of k.

4 General Graphs

For general graphs, we first look at the case $k = n$. We saw that in cycles there can be only two different n-particle stable sets. One requires an even n and is shown in Figure 1(b), while the other applies to an arbitrary $n \geq 3$ and is shown in Figure 2(a) with $r = 1$. The stable set in Figure 1(b) can be viewed in the following way. The set of edges E of an even-length cycle is partitioned into two perfect matchings M_1 and M_2. For any configuration in this stable set, either all vertex pointers are set onto the edges in M_1 or all of them are set onto the edges in M_2. We describe now a generalization of such stable sets to graphs with higher vertex degrees, considering perfect matchings as well as 2-factors (collections of vertex-disjoint cycles covering all vertices). We show first that n-particle stable sets can exist only in regular graphs.

Theorem 3. *If a connected n-vertex graph has an n-particle stable set, then the graph must be regular.*

Proof. Consider a connected n-vertex graph $G = (V, E)$ which has an n-particle stable set. Let C be a circuit in the Eulerian decomposition for this stable set. In each step one particle leaves each vertex, so in each step each vertex pointer advances to the next arc. Therefore, if a particle x assigned to C passes in the

current step through an arc (v, w), then the next particle on C will pass through (v, w) in exactly $\deg(v)$ steps, where $\deg(v)$ is the degree of v in G. This means that the particles assigned to C must be equally spaced around C, with distance $\deg(v)$ between the consecutive particles. Since v is an arbitrary vertex on C, all vertices on C must have the same degree. Thus if two circuits share a vertex, then all vertices on these two circuits have the same degree. Since the graph is connected, all vertices must have the same degree.

Lemma 10. *A connected d-regular n-vertex graph has an n-particle stable configuration if and only if the set of arcs $\overrightarrow{E}$ can be partitioned into sets H_1, $H_2, \ldots, H_d$, such that each H_i is a collection of vertex-disjoint simple arc-cycles.*

Proof. If there is an n-particle stable configuration S in a d-regular graph, then denote by S_i, for $i = 1, 2, \ldots, d$, the configuration at the beginning of step i, starting form configuraton $S = S_1$. Let H_i be the set of pointer arcs in configuration S_i. Each H_i is a collection of vertex-disjoint simple arc-cycles (the movement of the n particles in a given step defines a one-to-one mapping on V) and each arc belongs to exactly one H_i (from the rotor-router property).

Conversely, if H_1, $H_2, \ldots, H_d$ are collections of vertex-disjoint simple arc-cycles and these collections partition $\overrightarrow{E}$, then the d arcs outgoing from any vertex belong to different collections. For each vertex v, set the order of the arcs outgoing from v so that the i-th arc is the arc belonging to H_i, and initialize the vertex pointer to the first arc. These orders of arcs, the vertex pointers, and the assignment of one particle to each vertex define an n-vertex stable configuration.

Lemma 11. *If the edges of a connected regular graph can be partition into 2-factors and perfect matchings, then this graph admits an n-particle stable configuration.*

Proof. Each perfect matching defines one vertex-disjoint collection of simple arc-cycles covering V: each edge of the matching defines one two-arc cycle. Each 2-factor gives two vertex-disjoint collections of arc-cycles: for each cycle in the 2-factor, one orientation of this cycle is included in one collection and the other orientation in the other collection.

Corollary 4. *Let d be a positive integer. Each n-vertex, $(2d)$-regular graph admits an n-particle stable configuration. Each n-vertex, $(2d+1)$-regular graph which has a perfect matching admits an n-particle stable configuration. These graphs include all $(2d+1)$-regular $(2d)$-connected graphs.*

Proof. Petersen's 2-factor theorem says that every regular graph of even degree has a 2-factor, so (by iterating this theorem) it can be partitioned into 2-factors (see [15]). If a $(2d+1)$-regular graph has a perfect matching, then the edges which are not in this perfect matching form a $(2d)$-regular graph.

Petersen's matching theorem says that every 3-regular, 2-connected graph has a perfect matching. Babler's generalization of this theorem says that every $(2d+1)$-regular, $(2d)$-connected graph has a perfect matching (see [15]).

We now consider the case when $k < n$. In cycles, k-particle stable configurations exist only if n is a multiple of k, or a multiple of $k/2$ for an even $k \geq 4$. Thus in a cycle of prime length any initial configuration leads to full coalescing. There are no similar strong conditions for general graphs. Actually, if we allow vertices of degree 3 or higher, then the coalescence seems to depend more on the structure of the graph than on the primality of n or m. As an example, we show that for each sufficiently large m (which can be prime) and each $2 \leq k \leq \sqrt[3]{m/6}$, there is a connected graph with m edges which admits a k-particle stable configuration. Both the number of edges and the number of nodes in this example can be co-prime with the number of particles.

Our example is illustrated in Figure 5. The set of edges E is partitioned into $k+2$ components: a tree T with $k+1$ leaves $r_0, r_1, \ldots, r_k$, and at most $k-1$ internal vertices, each of degree at least 3, and vertex-disjoint connected sub-graphs $H_0, H_1, \ldots, H_k$. Sub-graph H_i shares vertex r_i with T and has either $\lfloor h \rfloor$ or $\lceil h \rceil$ edges, where $h = (m - |T|)/(k+1)$ and $|T| \leq 2k-1$ is the number of edges in T. We now show a k-particle stable configuration in this graph. Fix an Eulerian cycle C of the whole set $\overrightarrow{E}$. The arcs of H_i form one segment of C, which is an Eulerian circuit C_i of the arcs of H_i. If we remove all circuits C_i from C, then the remaining arcs form an Eulerian tour of T. The numbering $r_0, r_1, \ldots, r_k$ of leaves of T is consistent with the reverse order of this Eulerian tour of T. For each component H_i, we set the cyclic orders of arcs and the vertex pointers in such a way that one particle starting from vertex r_i would first follow the whole circuit C_i before entering tree T. The positions of the pointers are not final yet; they will be adjusted. The (cyclic) order $(v, w_1), \ldots, (v, w_{\deg(v)})$ of the arcs outgoing from an internal vertex v in T is consistent with the numbering of the leaves of T: if arcs $(v, w_1), \ldots, (v, w_j)$ lead to leaves $r_{i_1}, \ldots, r_{i_j}$, respectively, then $r_{i_1} < r_{i_2} < \cdots < r_{i_j}$ (the anti-clockwise order in Figure 5). The pointers at the internal vertices in T are set in the direction of vertex r_k.

The final stage of our construction of a stable configuration is the placement of the k particles $x_0, x_1, \ldots, x_{k-1}$ and the adjustment of the vertex pointers. All particles will be following the Eulerian circuit C and we show their initial positions in relation to this circuit. We place particle x_0 at vertex r_0 and change the vertex pointer at r_0 to the arc (r_0, p) of the tree T. This will be the next arc on C taken by x_0. We place particle x_1 on C at distance either $\lfloor g \rfloor$ or $\lceil g \rceil$ arcs *behind* particle x_0, where $g = 2m/k$. Generally, we place particles $x_1, x_2, \ldots, x_k$ so that each distance from x_i to x_{i-1} (including from x_0 to x_{k-1}) is either $\lfloor g \rfloor$ or $\lceil g \rceil$. Thus the distance from x_i to x_0 is between $i\lfloor g \rfloor$ and $i\lceil g \rceil$. The values h and g and the assumption that $k \leq \sqrt[3]{m/6}$ imply that for each $i = 1, 2, \ldots, k-1$, $i\lfloor g \rfloor \geq 2(|H_0| + |H_1| + \cdots + |H_{i-1}| + |T|)$ and $i\lceil g \rceil \leq 2(|H_0| + |H_1| + \cdots + |H_i|)$. This means that particle x_i is in H_i and H_k is empty (does not have any particle). Finally, for each $i = 1, 2, \ldots, k-1$, we adjust the vertex pointers in H_i by simulating the rotor-router movement of a "ghost" particle from r_i to the position of particle x_i. With this adjustment of vertex pointers in H_i, particle x_i will complete traversing circuit C_i (the traversing started by the ghost particle) and

then will enter tree T (assuming no interference from other particles). This completes the construction of a stable configuration.

Starting from the constructed configuration, the particles will move according to the following pattern. First particle x_0 moves to r_k along the $r_0 - r_k$ path in T in $O(k)$ steps, while the other particles move inside their initial H sub-graphs. Then particle x_1 completes the traversing of H_1, arrives at vertex r_1 and is ready to enter tree T. This completes the first phase and at this point, we have a configuration similar to the initial configuration, but now H_0 is empty. In the next phase, first particle x_1 moves to r_0 along the $r_1 - r_0$ path in T, while the other particles move inside their current H sub-graphs and x_2 reaches r_2. In the subsequent phases, particle x_2 moves from H_2 to H_1, then particle x_3 moves from H_3 to H_2, and so on. After $k(k+1)$ phases, the system is back in the initial configuration. It can be shown that no two particles will be at the same time in T or in the same H component, so no two particles ever collide.

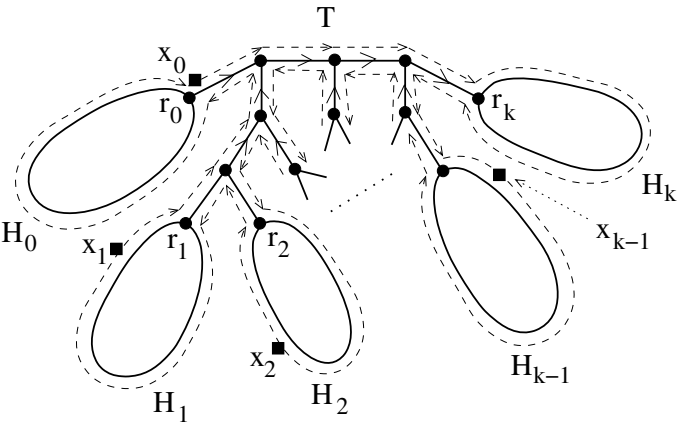

Fig. 5. A graph with m edges and a stable k-particle configuration, where $k = \Theta(m^{1/3})$.

References

1. Aldous, D., Fill, J.A.: Reversible markov chains and random walks on graphs 2002. Unfinished monograph, recompiled (2014). `http://www.stat.berkeley.edu/~aldous/RWG/book.html`
2. Aldous, D.J.: Meeting times for independent markov chains. Stochastic Processes and their Applications 38(2), 185–193 (1991)
3. Alon, N., Avin, C., Koucky, M., Kozma, G., Lotker, Z., Tuttle, M.R.: Many random walks are faster than one. In: Proc. 20th Annual Symposium on Parallelism in Algorithms and Architectures, SPAA 2008, pp. 119–128. ACM (2008)
4. Bampas, E., Gąsieniec, L., Hanusse, N., Ilcinkas, D., Klasing, R., Kosowski, A.: Euler tour lock-in problem in the rotor-router model. In: Keidar, I. (ed.) DISC 2009. LNCS, vol. 5805, pp. 423–435. Springer, Heidelberg (2009)
5. Bampas, E., Gasieniec, L., Klasing, R., Kosowski, A., Radzik, T.: Robustness of the rotor-router mechanism. In: Abdelzaher, T., Raynal, M., Santoro, N. (eds.) OPODIS 2009. LNCS, vol. 5923, pp. 345–358. Springer, Heidelberg (2009)

6. Bhatt, S.N., Even, S., Greenberg, D.S., Tayar, R.: Traversing directed eulerian mazes. J. Graph Algorithms Appl. 6(2), 157–173 (2002)
7. Chalopin, J., Das, S., Gawrychowski, P., Kosowski, A., Labourel, A., Uznanski, P.: Lock-in problem for parallel rotor-router walks. CoRR, abs/1407.3200 (2014)
8. Cooper, C., Elsässer, R., Ono, H., Radzik, T.: Coalescing random walks and voting on connected graphs. SIAM J. Discrete Math. 27(4), 1748–1758 (2013)
9. Cooper, C., Frieze, A.M., Radzik, T.: Multiple random walks in random regular graphs. SIAM J. Discrete Math. 23(4), 1738–1761 (2009)
10. Dereniowski, D., Kosowski, A., Pajak, D., Uznanski, P.: Bounds on the cover time of parallel rotor walks. In: 31st International Symposium on Theoretical Aspects of Computer Science, STACS 2014, pp. 263–275 (2014)
11. Efremenko, K., Reingold, O.: How well do random walks parallelize? In: Dinur, I., Jansen, K., Naor, J., Rolim, J. (eds.) APPROX and RANDOM 2009. LNCS, vol. 5687, pp. 476–489. Springer, Heidelberg (2009)
12. Elsässer, R., Sauerwald, T.: Tight bounds for the cover time of multiple random walks. Theor. Comput. Sci. 412(24), 2623–2641 (2011)
13. Israeli, A., Jalfon, M.: Token management schemes and random walks yield self-stabilizing mutual exclusion. In: Proceedings of the Ninth Annual ACM Symposium on Principles of Distributed Computing, PODC 1990, pp. 119–131. ACM (1990)
14. Kosowski, A., Pająk, D.: Does adding more agents make a difference? A case study of cover time for the rotor-router. In: Esparza, J., Fraigniaud, P., Husfeldt, T., Koutsoupias, E. (eds.) ICALP 2014, Part II. LNCS, vol. 8573, pp. 544–555. Springer, Heidelberg (2014)
15. Lovász, L., Plummer, D.: Matching Theory. AMS Chelsea Publishing Series. American Mathematical Soc. (2009)
16. Oliveira, R.: On the coalescence time of reversible random walks. Trans. Amer. Math. Soc. 364, 2109–2128 (2012)
17. Priezzhev, V.B., Dhar, D., Dhar, A., Krishnamurthy, S.: Eulerian walkers as a model of self-organized criticality. Phys. Rev. Lett. 77, 5079–5082 (1996)
18. Wagner, I.A., Lindenbaum, M., Bruckstein, A.M.: Smell as a computational resource - A lesson we can learn from the ant. In: ISTCS, pp. 219–230 (1996)
19. Wagner, I.A., Lindenbaum, M., Bruckstein, A.M.: Distributed covering by ant-robots using evaporating traces. IEEE T. Robotics and Automation 15(5), 918–933 (1999)
20. Yanovski, V., Wagner, I.A., Bruckstein, A.M.: A distributed ant algorithm for efficiently patrolling a network. Algorithmica 37(3), 165–186 (2003)