Deterministic Random Walks on the Two-Dimensional Grid

Benjamin Doerr and Tobias Friedrich

Max-Planck-Institut für Informatik, Saarbrücken, Germany

Abstract. Deterministic and randomized balancing schemes are used to distribute workload evenly in networks. In this paper, we compare two very general ones: The random walk and the (deterministic) Propp machine. Roughly speaking, we show that on the two-dimensional grid, the Propp machine always has the same number of tokens on a node as does the random walk in expectation, apart from an additive error of less than eight. This constant is independent of the total number of tokens and the runtime of the two processes. However, we also show that it makes a difference whether the Propp machine serves the neighbors in a circular or non-circular order.

1 Introduction

Given an arbitrary graph, a *random walk* is a path which begins at a given starting point and chooses the next node from the set of its current neighbors uniformly at random. Random walks have been used to model a wide variety of processes in economics (share prices), physics (Brownian motion of molecules in gases and liquids), medicine (cascades of neurons firing in the brain), and mathematics (estimations and modeling of gambling). In computer science, they are the heart of many randomized algorithms.

Jim Propp suggested the following quasirandom analogue to the random walk. The study of quasirandom approaches is motivated by the experience that in many applications they proved to be superior to random ones. Propp's *rotor-router model*, which we prefer to call *Propp machine*, is a simple deterministic process. Each vertex is equipped with a "rotor" which points to one of its neighbors. Instead of leaving a vertex in a random direction, the Propp walk follows the direction of the rotor. Afterwards, the rotor is updated to point to a new neighbor. For this, it follows a fixed cyclic permutation of the adjacent vertices.

The rotors ensure a very high degree of fairness. If a Propp walk visits some vertex v exactly k times, then for each neighbor w of v it does the passage $v \to w$ either $\lfloor k/\deg(v) \rfloor$ or $\lceil k/\deg(v) \rceil$ times. While in the random walk all these numbers are $k/\deg(v)$ in expectation, with high probability they deviate from that by $\Theta(\sqrt{k/\deg(v)})$. Therefore, in this respect the Propp machine is a better simulation for the "expected random walk" than the random walk itself.

The Propp machine found considerable attention recently [2, 3, 5, 6]. In this paper, we compare the two models with respect to their balancing behavior, another well-known application of random walks in computer science. In this setting, each vertex holds a number of chips. These chips simultaneously and in a synchronized manner perform a walk (random or Propp, depending on the model). Clearly, we expect both model to reduce imbalances between the occupation of vertices in each time step. As an idealized balancing scheme, we also regard the *linear machine*. Here, we allow fractional chips, and in each time step each vertex sends out exactly the same number (possibly non-integral) of chips to each neighbor. Clearly, the linear machine describes what the random walk does *in expectation*.

A well-studied problem is *balancing the workload* in a massively parallel computer or distributed network [9]. Each node/processor initially has a collection of tasks and the object is to balance the number of tasks at each node by moving them to neighboring nodes. Hence, processors are modeled as vertices of an undirected connected graph and links between them as edges. Jobs are modeled as unit-size tokens. There are two models prevalent in the literature: the dimension exchange and the diffusion paradigm. In both models each node's decisions are based only on local knowledge.

The intuition behind *diffusion* is that the number of tokens a processor sends to its neighbor is proportional to the differential between the two processors and some constant depending on the connecting edge [4]. A standard choice for these constants is uniform diffusion, in which each processor simply averages the loads of its neighbors at each step. This is usually modeled by allowing fractional tasks and ignoring the roundings to whole tasks at each local balancing step. However, the resulting deviations can become quite large, as was shown in [8]. To quantify the deviation between the fractional problem and the (real-world) integer problem is an important step in understanding diffusive load-balancing.

Dimension exchange refers to (periodic) balancing circuits [7]. This model is particularly suited to single-port architectures, where processors can only communicate with one of its d neighbors at a time. It decomposes the network in a sequence $M_1, \ldots, M_d$ of perfect matchings and adds a balancer at each edge. The balancer is a simple toggling device pointing to one of the incident nodes. Its purpose is to balance the flow of chips along the wires. Each balancing round consists of d steps, one for each matching. In step k, two nodes i, j which are matched in M_k balance their loads x_i, x_j as closely as possible, i.e., their loads become $\lfloor \frac{x_i+x_j}{2} \rfloor$ and $\lceil \frac{x_i+x_j}{2} \rceil$ with the excess chip following the balancer of the edge $\{i, j\}$.

Both models raise the question how well such quasirandom approaches simulate the random one, in particular, how close they get to the idealized linear approach (the random approach "in expectation").

In order to stick not too closely to a particular workload balancing approach for a particular distributed network, we analyze what we think is a sufficiently general but simple model. We compare the classical random walk (respectively the linear machine describing its expectation) with the Propp machine. Hence we do not have weights attached to the edges. As underlying graph we choose the two-dimensional infinite grid. Clearly, infinity is not a realistic assumption in a computer network setting. However, since in finite time chips can walk only a finite distance, the behavior we detect also occurs on finite grids. Hence the infinity assumption is rather used to get rid of some extra technicalities. Also, we assume that our results can be extended to settings with weights attached to the edges, as well as to other graph topologies or a setting where chips may also stay on a vertex. We would like to stress that the focus of our research is more fundamental than oriented to direct applicability. We do feel, however, that the same questions for a particular balancing setting are highly relevant from the view-point of application. Also, we are convinced that our methods can be applied there in an analogous manner.

To measure the difference between the two models, we estimate the maximal difference of the number of chips at any position and time in the Propp model compared to the number in the linear model if both models start with the same initial configuration. Apart from a technicality, which we defer to Section 2, Cooper and Spencer [2] proved for all grids $\mathbb{Z}^d$ that this discrepancy can be bounded by a constant c_d, which only depends on the dimension. In particular, c_d is independent of the initial configuration, the runtime, and the cyclic permutation of the cardinal directions (*rotor sequence*). For the graph being the infinite path, Cooper et al. [3] showed that this constant is $c_1 \approx 2.3$. They further proved that the discrepancy at the origin is maximized if each position sends exactly one odd chip at a certain time in the direction of the origin.

In this paper, we continue this work and rigorously analyze the Propp machine on the two-dimensional grid $\mathbb{Z}^2$. In comparison to the one-dimensional case, a number of new effects appear. In particular, the order in which the four neighbors are served now makes a difference. We prove $c_2 \approx 7.8$ for circular (i.e., clockwise or counterclockwise) rotor sequences and $c_2 \approx 7.3$ for all other rotor sequences. This is the first paper which regards the influence of these rotor sequences. We also precisely characterize the respective worst-case configurations. In particular, we prove that the worst-case is already reached when each position sends at most nine odd chips at at most three different times.

2 Preliminaries

To simplify the calculations, we rotate the grid by 45° and consider neighbors in directions $\textsc{dir} := \{\nearrow, \searrow, \swarrow, \nwarrow\}$. Note that by this, we only allow chips on positions $\mathbf{x} = \binom{x_1}{x_2}$ with $x_1 \equiv x_2 \pmod 2$. Since this model is isomorphic to the standard two-dimensional grid model with neighbors $\{\uparrow, \rightarrow, \downarrow, \leftarrow\}$, our result also holds for the standard model.

First, we fix some notation to describe chips on the grid. For $\mathbf{x} \in \mathbb{Z}^2$ and $t \in \mathbb{N}_0$, $\mathbf{x} \sim t$ denotes that $x_1 \equiv x_2 \equiv t \pmod 2$ and $\mathbf{x} \sim \mathbf{y}$ denotes that $x_1 \equiv x_2 \equiv y_1 \equiv y_2 \pmod 2$. A position $\mathbf{x}$ is called "even" or "odd" if $\mathbf{x} \sim 0$ or $\mathbf{x} \sim 1$, respectively. A configuration is called "even" or "odd" if all chips are at an even or all at an odd positions, respectively.

As pointed out in the introduction, there is one limitation without which neither the results of [2, 3] nor our results hold. Note that since $\mathbb{Z}^2$ is a bipartite graph, chips that start on *even* positions never mix with those starting on *odd* positions. It looks like we are playing two games at once. However, this is not true, because chips of different parity may affect each other through the rotors. Within each game the number of chips send in the four directions is not balanced at each position. One can cleverly arrange piles of off-parity chips to reorient rotors and steer them away from random walk simulation. We therefore require the starting configuration to have chips only on *one* parity. Without loss of generality, we consider only even starting configurations.

A random walk on $\mathbb{Z}^2$ can be described nicely by its probability density. By $H(\mathbf{x}, t)$ we denote the probability that a chip from location $\mathbf{x}$ arrives at the origin after t random steps (at time t) in a simple random walk. On a grid as defined above, this is

$$H(\mathbf{x}, t) = 4^{-t}\binom{t}{(t+x_1)/2}\binom{t}{(t+x_2)/2}$$

for $\mathbf{x} \sim t$ and $\|\mathbf{x}\|_\infty \leq t$, and $H(\mathbf{x}, t) = 0$ otherwise.

The order, in which the four neighbors in directions $\textsc{dir}$ are served, has a significant impact on the discrepancy between Propp and linear machine. We use the same rotor sequence for all positions and describe it by a cyclic function $\textsc{next}\colon \textsc{dir} \to \textsc{dir}$. Implicitly in the following notations, we fix the rotor sequence as well as the starting configuration. That is, the number of chips on vertices and rotor directions. Let $f(\mathbf{x}, t)$ denote the number of chips and $\textsc{arr}(\mathbf{x}, t)$ the direction of the arrow at position $\mathbf{x}$ after t steps of the Propp machine. Note that with this we can determine the resulting arrow after one Propp step via $\textsc{arr}(\mathbf{x}, t+1) = \textsc{next}^{f(\mathbf{x},t)}\big(\textsc{arr}(\mathbf{x}, t)\big)$.

Let $E(\mathbf{x}, t)$ denote the expected number of chips at location $\mathbf{x}$ after a random walk of all chips for t steps. In the proofs, we also need the following mixed notation. By $E(\mathbf{x}, t_1, t_2)$ we denote the expected number of chips at location x after first performing t_1 Propp and then $t_2 - t_1$ random walk steps.

3 Parity-Forcing Theorem

For a deterministic process like the Propp machine, it is obvious that the initial configuration (that is, the position of each chip and the direction of each rotor), determines all subsequent configurations. The following theorem shows a partial converse, namely that (roughly speaking) we may prescribe the number of chips modulo 4 on all vertices at all times and still find an initial configuration leading to such a game. An analogous result for the one-dimensional Propp machine has been shown in [3].

Theorem 1 (Parity-forcing Theorem). *For any fixed rotor sequence, any initial position of the arrows and any* $\pi\colon \mathbb{Z}^2 \times \mathbb{N}_0 \to \{0, 1, 2, 3\}$ *with* $\pi(\mathbf{x}, t) = 0$ *for all* $\mathbf{x} \not\sim t$*, there is an initial even configuration* $f(\mathbf{x}, 0)$*,* $\mathbf{x} \in \mathbb{Z}^2$ *that results in a game with* $f(\mathbf{x}, t) \equiv \pi(\mathbf{x}, t) \pmod 4$ *for all* $\mathbf{x}$ *and* t*.*

The proof is based on the observation that a pile of 4^t chips splits evenly t times. The details of this proof (and all proofs coming) are deferred to the full version of this paper.

4 The Basic Method

In this section we derive the main equations to compare Propp and linear machine based on the number of chips on a single vertex. We are interested in bounding the discrepancies $f(\mathbf{x},t) - E(\mathbf{x},t)$ for all vertices $\mathbf{x}$ and all times t. Since we aim at bounds independent of the starting configuration, it suffices to regard the vertex $\mathbf{x} = \mathbf{0}$. With

$$\begin{aligned} E(\mathbf{0},0,t) &= E(\mathbf{0},t), \\ E(\mathbf{0},t,t) &= f(\mathbf{0},t), \end{aligned}$$

we get

$$f(\mathbf{0},t) - E(\mathbf{0},t) \;=\; \sum_{s=0}^{t-1} \left(E(\mathbf{0},s+1,t) - E(\mathbf{0},s,t)\right). \tag{1}$$

By $\textsc{rem}_s^{(j)}$ we denote the set of positions that are occupied by k chips with $k \equiv j \pmod 4$ at time s. Note that if a position contains four chips, then these four chips behave identically on the Propp and linear machine. With this, we obtain

$$\begin{aligned} &E(\mathbf{0},s+1,t) - E(\mathbf{0},s,t) \qquad\qquad (2)\\ &= \sum_{\mathbf{x}\in\textsc{rem}_s^{(1)}} \big(H(\mathbf{x}+\textsc{arr}(\mathbf{x},s),t-s-1) - H(\mathbf{x},t-s)\big) \\ &\quad + \sum_{\mathbf{x}\in\textsc{rem}_s^{(2)}} \big(H(\mathbf{x}+\textsc{arr}(\mathbf{x},s),t-s-1) \\ &\qquad\qquad +H(\mathbf{x}+\textsc{next}(\textsc{arr}(\mathbf{x},s)),t-s-1) \\ &\qquad\qquad -2H(\mathbf{x},t-s)\big) \\ &\quad + \sum_{\mathbf{x}\in\textsc{rem}_s^{(3)}} \big(H(\mathbf{x}+\textsc{arr}(\mathbf{x},s),t-s-1) \\ &\qquad\qquad +H(\mathbf{x}+\textsc{next}(\textsc{arr}(\mathbf{x},s)),t-s-1) \\ &\qquad\qquad +H(\mathbf{x}+\textsc{next}^2(\textsc{arr}(\mathbf{x},s)),t-s-1) \\ &\qquad\qquad -3H(\mathbf{x},t-s)\big). \end{aligned}$$

We now regard single chips and define $s_i(\mathbf{x}) := \min\left\{u \geq 0 \,|\, i < \sum_{t=0}^{u} f(\mathbf{x},t)\right\}$ for all $i \in \mathbb{N}_0$, i.e., at time $s_i(\mathbf{x})$ the location $\mathbf{x}$ is occupied by its i-th chip (counting from 0). With

$$\textsc{inf}(\mathbf{x},\mathbf{A},t) := H(\mathbf{x}+\mathbf{A},t-1) - H(\mathbf{x},t)$$

for $\mathbf{x} \sim t$ and $\textsc{inf}(\mathbf{x},\mathbf{A},t) = 0$ otherwise, we denote the influence of position $\mathbf{x}$ with the arrow pointing to $\mathbf{A}$ at time t to the discrepancy between Propp and linear machine. A simple calculation yields

$$\textsc{inf}(\mathbf{x},\mathbf{A},t) = \left((A_1x_1 \cdot A_2x_2)t^{-2} - (A_1x_1 + A_2x_2)t^{-1}\right) H(\mathbf{x},t).$$

With these notations, Equations (1) and (2) give

$$f(\mathbf{0},T) - E(\mathbf{0},T) = \sum_{\mathbf{x}\in\mathbb{Z}^2}\sum_{i\geq 0} \textsc{inf}(\mathbf{x}, \textsc{next}^i(\textsc{arr}(\mathbf{x},0)), T - s_i(\mathbf{x})). \qquad (3)$$

This is the main equation, which will be examined in the remainder. It shows that the discrepancy is just the sum of the contributions

$$\textsc{con}(\mathbf{x}) := \sum_{i\geq 0} \textsc{inf}(\mathbf{x}, \textsc{next}^i(\textsc{arr}(\mathbf{x},0)), T - s_i(\mathbf{x}))$$

at all positions $\mathbf{x}$. This is a very important observation since this allows us to examine each position $\mathbf{x}$ separately.

5 The Modes of $\textsc{inf}(\mathbf{x}, \mathbf{A}, t)$

In the previous section, we expressed the discrepancy as a sum of certain influences $\textsc{inf}(\mathbf{x}, \mathbf{A}, t)$. We now analyze $\textsc{inf}(\mathbf{x}, \mathbf{A}, t)$. Let $X \subseteq \mathbb{R}$. We call a mapping $f\colon X \to \mathbb{R}$ *unimodal*, if there is an $m \in X$ such that $f|_{x\leq m}$ as well as $f|_{x\geq m}$ are monotone. We call a mapping $f\colon X \to \mathbb{R}$ *bimodal*, if there are $m_1, m_2 \in X$ such that $f|_{x\leq m_1}$, $f|_{m_1\leq x\leq m_2}$, and $f|_{m_2\leq x}$ are monotone. We call a mapping $f\colon X \to \mathbb{R}$ *strictly bimodal*, if it is bimodal, but not unimodal.

Unimodal functions are popular in optimization and probability theory. The probability $H(\mathbf{x}, t)$ that a chip from the origin arrives at location $\mathbf{x}$ at time t in a simple random walk is unimodal in t. For our purposes it is important that $\textsc{inf}(\mathbf{x}, \mathbf{A}, t)$ is bimodal. To prove this, we need Descartes' Rule of Signs, which can be found in [1]. With this, we are now well equipped to characterize $\textsc{inf}(\mathbf{x}, \mathbf{A}, t)$ precisely.

Lemma 2. *For all* $\mathbf{x}$ *and* $\mathbf{A}$*,* $\textsc{inf}(\mathbf{x}, \mathbf{A}, t)$ *is bimodal in* t*. It is strictly bimodal in* t *if and only if (i)* $\|\mathbf{x}\|_\infty > 6$ and *(ii)* $-A_1x_1 > A_2x_2 > (-A_1x_1 + 1)/2$ *or* $-A_2x_2 > A_1x_1 > (-A_2x_2 + 1)/2$.

We will also need the extrema of certain sums of $\textsc{inf}$'s. The following lemma shows that

$$\textsc{inf2}(\mathbf{x}, \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, t) := \textsc{inf}(\mathbf{x}, \mathbf{A}^{(1)}, t) + \textsc{inf}(\mathbf{x}, \mathbf{A}^{(2)}, t)$$

has even nicer properties than $\textsc{inf}$ itself. In particular, $\textsc{inf2}(\mathbf{x}, \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, t)$ is never strictly bimodal in t for $\mathbf{A}^{(1)} \neq \mathbf{A}^{(2)}$.

Lemma 3. *For all* $\mathbf{x}$ *and* $\mathbf{A}^{(1)} \neq \mathbf{A}^{(2)}$*,* $\textsc{inf2}(\mathbf{x}, \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, t)$ *is unimodal in* t*.*

6 Worst-Case Behavior

In Section 4 we derived that for a fixed initial configuration, the single vertex discrepancy is the sum of the contributions

$$\textsc{con}(\mathbf{x}) := \sum_{i\geq 0} \textsc{inf}(\mathbf{x}, A^{(i)}, t_i)$$

of all positions $\mathbf{x}$ with $A^{(i)} := \textsc{next}^i(\textsc{arr}(\mathbf{x},0))$ and $t_i := T - s_i(\mathbf{x})$. In this section we determine for each rotor sequence and each position $\mathbf{x}$ the maximum of $\textsc{con}(\mathbf{x})$ for all initial configurations. We denote this by $\textsc{maxcon}(\mathbf{x})$. Hence the sum of all $\textsc{maxcon}(\mathbf{x})$ gives an upper bound for the single vertex discrepancy. Due to the parity-forcing Theorem 1, there is also an initial configuration which sends from all locations $\mathbf{x}$ (apart from multiples of four) exactly the number of chips from $\mathbf{x}$ as the configurations with $\textsc{con}(\mathbf{x}) = \textsc{maxcon}(\mathbf{x})$. Thus, the upper bound is tight:

$$f(\mathbf{0},T) - E(\mathbf{0},T) = \sum_{\mathbf{x}\in\mathbb{Z}^2} \textsc{maxcon}(\mathbf{x}). \tag{4}$$

We now fix a rotor sequence and a position $\mathbf{x}$ and examine $\textsc{maxcon}(\mathbf{x})$. Lemmas 2 and 3 prove that $\textsc{inf}(\mathbf{x},\mathbf{A},t)$ and $\textsc{inf}(\mathbf{x},\mathbf{A}^{(1)},t)+\textsc{inf}(\mathbf{x},\mathbf{A}^{(2)},t)$ are bimodal in t. We observe that sending a chip in each direction at the same time does not change $\textsc{con}(\mathbf{x})$. For all t we have

$$\sum_{\mathbf{A}\in\{\nearrow,\searrow,\swarrow,\nwarrow\}} \textsc{inf}(\mathbf{x},\mathbf{A},t) = 0. \tag{5}$$

This shows that all $\sum_i \textsc{inf}(\mathbf{x},\mathbf{A}^{(i)},t)$ are bimodal in t. Since $\textsc{inf}(\mathbf{x},\mathbf{A},0) = \lim_{t\to\infty} \textsc{inf}(\mathbf{x},\mathbf{A},t) = 0$ for all $\mathbf{A}$, each $\sum_i \textsc{inf}(\mathbf{x},\mathbf{A}^{(i)},t)$ has at most two extremal times. The set of all extremal times of all $\sum_i \textsc{inf}(\mathbf{x},\mathbf{A}^{(i)},t)$ can be defined as follows.

$$\begin{aligned}\textsc{ex}(\mathbf{x}) := & \bigcup_{\substack{\mathbf{A}^{(1)},\mathbf{A}^{(2)}\in\{\nearrow,\searrow,\swarrow,\nwarrow\} \\ \max_t \textsc{inf2}(\mathbf{x},\mathbf{A}^{(1)},\mathbf{A}^{(2)},t)>0}} \operatorname*{argmax}_t \textsc{inf2}(\mathbf{x},\mathbf{A}^{(1)},\mathbf{A}^{(2)},t) \;\cup \\ & \bigcup_{\substack{\mathbf{A}^{(1)},\mathbf{A}^{(2)}\in\{\nearrow,\searrow,\swarrow,\nwarrow\} \\ \min_t \textsc{inf2}(\mathbf{x},\mathbf{A}^{(1)},\mathbf{A}^{(2)},t)<0}} \operatorname*{argmin}_t \textsc{inf2}(\mathbf{x},\mathbf{A}^{(1)},\mathbf{A}^{(2)},t)\end{aligned}$$

with $\operatorname{argmax}_t f(t) := \max\{s \mid f(s) = \max_t f(t)\}$ and $\operatorname{argmin}_t f(t) := \max\{s \mid f(s) = \min_t f(t)\}$. Notice that

$$\begin{aligned}\textsc{inf}(\mathbf{x},\mathbf{A},t) &= \tfrac{1}{2}\textsc{inf2}(\mathbf{x},\mathbf{A},\mathbf{A},t), \\ \textstyle\sum_{i=0}^{1} \textsc{inf}(\mathbf{x},\mathbf{A}^{(i)},t) &= \textsc{inf2}(\mathbf{x},\mathbf{A}^{(0)},\mathbf{A}^{(1)},t), \text{ and} \\ \textstyle\sum_{i=0}^{2} \textsc{inf}(\mathbf{x},\mathbf{A}^{(i)},t) &= -\tfrac{1}{2}\textsc{inf2}(\mathbf{x},\mathbf{A}^{(3)},\mathbf{A}^{(3)},t).\end{aligned}$$

Lemmas 2 and 3 imply $|\textsc{ex}(\mathbf{x})| \leq 7$. By calculating the roots of the polynomials $p(\mathbf{x},\mathbf{A},t)$ and $p(\mathbf{x},\mathbf{A}^{(1)},t) + p(\mathbf{x},\mathbf{A}^{(2)},t)$ given in Lemma 2, it is easy to determine $\textsc{ex}(\mathbf{x})$. In Lemma 4 below we will show that there are configurations with $\textsc{con}(\mathbf{x}) = \textsc{maxcon}(\mathbf{x})$ where chips are only send from $\mathbf{x}$ at times $\textsc{ex}(\mathbf{x})$. This proof is based on the following blocking argument.

A *phase* (of $\mathbf{x}$) denotes a maximal period of time, in which all sums of $\textsc{inf}(\mathbf{x},\mathbf{A},t)$'s are monotonic. That is, the upper and lower limit of a phase is either an extremal time $t \in \textsc{ex}(\mathbf{x})$, or 0, or T. Note that we can assume $T > \max \textsc{ex}(\mathbf{x})$ and that the monotonicity is uniquely determined for all $\mathbf{A}$ in each phase.

A *block* denotes four consecutive chips, at (possibly different) times $t_j, \dots, t_{j+3}$ send from $\mathbf{x}$ within one phase such that $\sum_{i=j}^{j+k} \textsc{inf}(\mathbf{x}, \mathbf{A}^{(i)}, t)$ is monotonically increasing in t for all $k \in \{0, 1, 2\}$. By Equation (5), this is equivalent to $\sum_{i=j+k}^{j+3} \textsc{inf}(\mathbf{x}, \mathbf{A}^{(i)}, t)$ being monotonically decreasing in t for all $k \in \{1, 2, 3\}$.

A *block prefix* denotes $0 \leq \ell \leq 3$ consecutive chips at times $t_j, \dots, t_{j+\ell-1}$ send from $\mathbf{x}$ within one phase with $\sum_{i=j}^{j+k} \textsc{inf}(\mathbf{x}, \mathbf{A}^{(i)}, t)$ monotonically increasing in t for all $0 \leq k < \ell$. A *block suffix* denotes $0 \leq \ell \leq 3$ consecutive chips at times $t_j, \dots, t_{j+\ell-1}$ send from $\mathbf{x}$ within one phase with $\sum_{i=j+k}^{j+\ell-1} \textsc{inf}(\mathbf{x}, \mathbf{A}^{(i)}, t)$ monotonically decreasing in t for all $0 \leq k < \ell$.

To describe chips in a phase, we use $\rightarrow$ and $\leftarrow$ to denote chips send in arrow direction $\mathbf{A}^{(i)}$ whose $\textsc{inf}(\mathbf{x}, \mathbf{A}^{(i)}, t)$ is increasing or decreasing in t, respectively. With this notation, there are four types of blocks: $\rightarrow\leftarrow\leftarrow\leftarrow$, $\rightarrow\rightarrow\leftarrow\leftarrow$, $\rightarrow\rightarrow\rightarrow\leftarrow$, and $\rightarrow\leftarrow\rightarrow\leftarrow$. There are four important properties of blocks, which are shown easily:

- For all blocks, there is a common time t with $t_j \leq t \leq t_{j+3}$ such that $\sum_{i=j}^{j+3} \textsc{inf}(\mathbf{x}, \mathbf{A}^i, t_i) \leq \sum_{i=j}^{j+3} \textsc{inf}(\mathbf{x}, \mathbf{A}^i, t) = 0$. Hence, removing a block does not decrease $\textsc{con}(\mathbf{x})$.
- For all block suffixes and prefixes, there is a common time t with $t_j \leq t \leq t_{j+\ell-1}$ such that $\sum_{i=j}^{j+\ell-1} \textsc{inf}(\mathbf{x}, \mathbf{A}^i, t_i) \leq \sum_{i=j}^{j+\ell-1} \textsc{inf}(\mathbf{x}, \mathbf{A}^i, t)$. Hence, sending the ℓ chips of a block suffix or prefix at a common time t instead at times $t_j, \dots, t_{j+\ell-1}$ does not decrease $\textsc{con}(\mathbf{x})$.
- In each phase, the block type is uniquely determined by the monotonicity of $\textsc{inf}$ and $\textsc{inf2}$.
- Any sequence of chips sent within one phase, can be partitioned in a block suffix, zero or more blocks, and a block prefix.

Lemma 4. *There is an initial configuration with* $\textsc{con}(\mathbf{x}) = \textsc{maxcon}(\mathbf{x})$ *such that there are only chips send from* $\mathbf{x}$ *at times* $\textsc{ex}(\mathbf{x})$*.*

Lemma 4 shows that configurations with $\textsc{con}(\mathbf{x}) = \textsc{maxcon}(\mathbf{x})$ that send a minimal number of chips, only send chips from $\mathbf{x}$ at times $\textsc{ex}(\mathbf{x})$. With $\textsc{maxcon}_t(\mathbf{x})$ denoting the contribution at time $t \in \textsc{ex}(\mathbf{x})$, we obtain

$$\textsc{maxcon}(\mathbf{x}) = \sum_{t \in \textsc{ex}(\mathbf{x})} \textsc{maxcon}_t(\mathbf{x}).$$

The following lemma proves that $\textsc{maxcon}_t(\mathbf{x})$ is uniquely determined.

Lemma 5. $\textsc{maxcon}_t(\mathbf{x})$ *is uniquely determined by the monotonicity of* $\textsc{inf}$ *and* $\textsc{inf2}$ *in the two adjacent phases.*

For all $\mathbf{x}$ we can now characterize exactly the configuration with $\textsc{con}(\mathbf{x}) = \textsc{maxcon}(\mathbf{x})$ that sends the least number of chips. Note that by Equation (4),

$\sum_{\mathbf{x}} \textsc{maxcon}(\mathbf{x})$ is not only an upper bound for the single vertex discrepancy $f(\mathbf{0},T) - E(\mathbf{0},T)$, but also a lower bound. With the help of a computer, it is now easy to sum up over a large number of positions $\mathbf{x}$ and to calculate

$$\sum_{\|x\|_\infty \leq 800} \textsc{maxcon}(\mathbf{x}) \approx \begin{cases} 7.831 & \text{for circular rotor sequences} \\ 7.285 & \text{for other rotor sequences.} \end{cases}$$

Notice that these constants are just lower bounds for the single vertex discrepancy. To prove that they are upper bounds as well, we have to bound $E := \sum_{\|x\|_\infty > 800} \textsc{con}(\mathbf{x})$. Equation (3) yields

$$\begin{aligned} E \;\leq \sum_{\|x\|_\infty > 800} \Bigg(& \sum_{i \geq 0} -A_1^{(i)}(\mathbf{x}) \frac{x_1 H(\mathbf{x}, t - s_i(\mathbf{x}))}{t - s_i(\mathbf{x})} + \\ & \sum_{i \geq 0} -A_2^{(i)}(\mathbf{x}) \frac{x_2 H(\mathbf{x}, t - s_i(\mathbf{x}))}{t - s_i(\mathbf{x})} + \\ & \sum_{i \geq 0} A_1^{(i)}(\mathbf{x}) A_2^{(i)}(\mathbf{x}) \frac{x_1 x_2 H(\mathbf{x}, t - s_i(\mathbf{x}))}{(t - s_i(\mathbf{x}))^2} \Bigg) \end{aligned} \tag{6}$$

for all times $t \in \mathbb{N}_0$. Note that, independent of the chosen rotor sequence, each of the sequences $(A_1^{(i)}(\mathbf{x}))_{i \geq 0}$, $(A_2^{(i)}(\mathbf{x}))_{i \geq 0}$, and $(A_1^{(i)}(\mathbf{x}) A_2^{(i)}(\mathbf{x}))_{i \geq 0}$ are either strictly or in groups of two alternating. To bound the alternating sums of Equation (6), we need the following elementary fact.

Lemma 6. *Let $t_0, \ldots, t_n \in X \subseteq \mathbb{R}$ such that $t_0 \leq \ldots \leq t_n$. Let $f \colon X \to \mathbb{R}$ be non-negative and unimodal. If $A^{(i)}$ is either strictly alternating or alternating in groups of two, then*

$$\left| \sum_{i=0}^{n} A^{(i)} f(t_i) \right| \;\leq\; 2 \max_{x \in X} f(x).$$

The following lemma shows that (in contrast to $\textsc{inf}$) the three summands of Equation (6) are indeed always unimodal.

Lemma 7. *$H(\mathbf{x},t)/t$ and $H(\mathbf{x},t)/t^2$ are unimodal functions in t. Denote their global maxima with $t_{\max}(\mathbf{x})$ and $t'_{\max}(\mathbf{x})$, respectively. Then, $(x_1^2 + x_2^2)/4 - 2 \leq t_{\max}(\mathbf{x}) \leq (x_1^2 + x_2^2)/4 + 1$ and $(x_1^2 + x_2^2)/6 - 1 \leq t'_{\max}(\mathbf{x}) \leq (x_1^2 + x_2^2)/6 + 2$.*

By bounding the infinite sums with definite integrals, and applying Lemmas 6 and 7 we get $E < 0.15$, which finally proves

$$f(\mathbf{0},T) - E(\mathbf{0},T) \;\approx\; \begin{cases} 7.8 & \text{for circular rotor sequences} \\ 7.3 & \text{for other rotor sequences.} \end{cases}$$

References

[1] G. E. Collins and A. G. Akritas. Polynomial real root isolation using descarte's rule of signs. In *SYMSAC '76: Proceedings of the third ACM symposium on Symbolic and algebraic computation*, pp. 272–275, New York, NY, USA, 1976. ACM Press.

[2] J. Cooper and J. Spencer. Simulating a random walk with constant error. *Combinatorics, Probability and Computing.* (Also available at arXiv:math.CO/0402323).

[3] J. Cooper, B. Doerr, J. Spencer, and G. Tardos. Deterministic random walks. In *ANALCO'06: Proceedings of the Workshop on Analytic Algorithmics and Combinatorics*, pp. 185–197, Philadelphia, PA, 2006. SIAM.

[4] G. Cybenko. Dynamic load balancing for distributed memory multiprocessors. *J. Parallel Distrib. Comput.*, 7(2):279–301, 1989.

[5] M. Kleber. Goldbug Variations. *The Mathematical Intelligencer*, 27(1), 2005.

[6] L. Levine and Y. Peres. The rotor-router shape is spherical. *The Mathematical Intelligencer*, 27(3):9–11, 2005.

[7] Y. Rabani, A. Sinclair, and R. Wanka. Local divergence of markov chains and the analysis of iterative load-balancing schemes. In *FOCS'98: Proceedings of the 39th Annual Symposium on Foundations of Computer Science*, pp. 694–705. IEEE Computer Society, 1998.

[8] R. Subramanian and I. D. Scherson. An analysis of diffusive load-balancing. In *SPAA*, pp. 220–225, New York, NY, USA, 1994. ACM Press.

[9] C.-Z. Xu, B. Monien, R. Lüling, and F. C. M. Lau. An analytical comparison of nearest neighbor algorithms for load balancing in parallel computers. In *IPPS*, pp. 472–479. IEEE Computer Society, 1995.