Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2010, Article ID 936195, 11 pages
doi:10.1155/2010/936195

Research Article

Interacting Urns Processes for Clustering of Large-Scale Networks of Tiny Artifacts

Pierre Leone[1] and Elad M. Schiller[2]

[1] *Computer Science Department, Centre Universitaire d'Informatique Battelle Bâtiment, University of Geneva, A route de Drize 7, 1227 Carouge, Geneva, Switzerland*
[2] *Distributed Computing and Systems Research Group, Department of Computing Science, Chalmers University of Technology and Göteborg University Rännvägen, 6B S-412 96 Göteborg, Sweden*

Correspondence should be addressed to Pierre Leone, pierre.leone@cui.unige.ch

Received 3 October 2008; Revised 31 January 2009; Accepted 30 May 2010

We analyze a distributed variation on the Pólya urn process in which a network of tiny artifacts manages the individual urns. Neighboring urns interact by repeatedly adding the same colored ball based on previous random choices. We discover that the process rapidly converges to a definitive random ratio between the colors in every urn. Moreover, the rate of convergence of the process at a given node depends on the global topology of the network. In particular, the same ratio appears for the case of complete communication graphs. Surprisingly, this effortless random process supports useful applications, such as clustering and computation of pseudo-geometric coordinate. We present numerical studies that validate our theoretical predictions.

1. Introduction

Designers of distributed algorithms often assume that each node is computationally powerful, capable of storing nontrivial amounts of data, and carrying out complex calculations. However, recent technological developments in wireless communications and microprocessors allow us to establish networks consisting of massive amounts of cheap and tiny artifacts that are tightly resource constrained. These networks of tiny artifacts are far more challenging than the traditional networks; on one hand, their relative scale is enormous, while on the other hand, each tiny artifact can only run a millicode that is aided by a miniature memory. Such limitations are not crippling if the system designer has a precise understanding of the tiny artifacts' computational power and local interaction rules from which global formation emerges.

1.1. Related Work. Urn processes (and more generally processes with reinforcements) are a tool for modeling stochastic processes that show emerging structures. Their aim is to analyze the global structure of a given population given the micromechanisms the particular entities are applying. The global structure reflects the ability of the entities to self-organize. Such models can explain the preferential-attachment model in small-world networks, the process of market sharing, interaction of biological entities, and so forth, based on given micromechanisms. We refer to [1, and references therein] for reinforcement processes, for models based on urn models see [2], and for the emergence of structure we refer to [3].

Tishby and Slonim [4] use random processes for network clustering. The authors use a Markov process and the clusters are built by considering the decay of mutual information. The information changes with time, and during an appropriate period, the mutual information is relevant for clustering. After this period, the mixing property of the Markov process destroys the emerged structure. The random processes we suggest converge so slowly that the mixing property is useful for tolerating various faults. Nevertheless, the process obtains the significant values rapidly.

Angluin et al. [5] define *urn automata* that include a state controller and an urn containing balls with a finite set of colors. Tiny artifacts can implement the urn automata with provable guarantees regarding their computational power and, by that, allowing the exact analyses of the interaction process among artifacts. This work aims at understanding the possibilities of the urn automata to emerging global formations using miniature algorithms and diminutive resources.

1.2. Our Contribution. In this paper our aim is to extend the classical urn process to a network of interacting urn processes and to analyze the emerging global formation, which is carried out by a low-level interaction process. The network of tiny artifacts repeats the following operations. Each urn has a multiset of black and white balls, initially one of each color. A given tiny artifact draws a ball from the urn with uniform distribution and returns to the urn the ball along with a new ball of the same color. (This is a classical Pólya urn [6].) The tiny artifact interacts with its neighbors on the network. Namely, after returning the two balls, the tiny artifact announces the balls' color to its neighbors, and the neighbors add a new ball of that color to their urns. The selection of the artifact drawing the ball from the urn is random and uniform (any artifact might be selected with the same probability).

Indeed, if we assume that, before drawing from the urn, the tiny artifacts wait for a random time with λ-exponential distribution, then globally, the tiny artifacts are going to be selected uniformly without any global synchronization. (The implementation of this assumption in distributed systems is considered in Section 4.) Notice that the algorithm steps are composed of the following operations: (1) an urn ball-drawing operation, (2) a ball announcement operation, and (3) an update of the urn by the drawing tiny artifact and its neighboring urns. Concurrent steps should interleave carefully; we require that at most one artifact among any set of neighbors is accessing its urn at a time and announces its ball drawing, and no neighboring artifact takes a step before all neighboring artifacts update their urns.

This work presents the following.

(i) First is an analysis of the interacting urns process, allowing us to demonstrate that the ratio between the numbers of black and white balls in every urn is converging and that these ratios, which are obtained locally, provide global information about the communication graph.

(ii) Second are the applications for large-scale networks of tiny artifacts include clustering and the computation of pseudogeometric coordinates. We present numerical studies that validate our theoretical predictions on the emerging global formations.

Self-stabilizing systems [7, 8] can recover after the occurrence of transient faults. These systems are designed to automatically regain their consistency from any starting state. The property of self-stabilizing facilitates the requirements of self-organization in ad hoc networks [9–11].

(iii) The interacting urn process analysis assumes undisturbed steps and that all urns are selected with the same probability. We explain how to implement the interacting urn process in a self-stabilizing manner (see [8]). We use existing self-stabilizing building block that facilitates the announcement of ball drawing on shared communication media.

1.3. Document Structure. We start by analyzing the process (Section 2), before presenting the applications (Section 3), and the implementation (Section 4). Lastly, we draw our conclusions (Section 5).

2. The Interacting Urns Process

There is no unified way of analyzing the behavior over time of the type of systems we consider in this paper. These systems usually cannot be understood with Markovian formalism. Indeed, the system behavior over time is Markovian but describing the process in this way leads to an intractable set of equations. Relative to these kinds of processes, which are characterized by path-dependence or (negative) reinforcement, we can find various ad hoc techniques. Many references are available from the survey [1]. We show that there is a connection between a time-based analysis of the interacting urns process and multitype branching process [12]. We use this connection and existing literature to that the evolution equations of the interacting urns processed.

We start by establishing firmly how the tiny artefacts randomly draw balls from their urn with a uniform probability. For the sake of presentation simplicity, we assume in Lemma 1 that within a single operation that takes no time, we can draw a ball and advertise it, that is, announce the ball's color to all neighboring urns. (In Section 4, we consider a possible implementation that does not require this assumption.)

Lemma 1. *We consider N artefacts which repeatedly wait for an λ-exponentially distributed random time independently of each other before proceeding to a random draw. Then, the probability that artefacts proceed to a random draw is uniform.*

Proof. A λ-exponential waiting time can be facilitated by the following p-persistence-like technique. Every artefact decides with probability $1 - \lambda \cdot h$ to wait for a period of h time units and not to draw a ball and with probability $\lambda \cdot h$ it decides to draw a ball and announce the ball's color to all neighboring urns. The λ-exponential distribution is obtained as $h \to 0$. It is easy to check that as $h \to 0$, the probability that two artefacts draw a ball simultaneously vanishes. This proves the lemma because during short period h time units, the artefacts independently decide to draw a ball or not. □

As a conclusion from Lemma 1, we say that the artefacts wait independently for a random exponentially distributed random time (with the same parameter λ) and then draw and announce a random ball from their urns. Let N be the number of artefacts composing the network. The state of the interacting urns process is described by the vector:

$$\mathbf{Z}(t) = \left(Z^1(t), \dots, Z^N(t)\right)^T \quad (1)$$

with $Z^i(t) = (b_t^i, w_t^i)^T$, where the index $i \in \{1, \dots, N\}$ refers to a particular artefact. (The ^{T} is the transpose operator which is used to denote the vectors column-wise, as usual.) (The presentation analysis uses urns' indices. However, the interacting urns are anonymous; that is, the urns do not know their indices and do not use unique identities in their computations or communications.) To simplify the presentation, we start by writing down the equations for the total population belonging to the urns $Z^i(t) = b_t^i + w_t^i$.

Every time the artefact owner or one of its neighboring artefacts proceeds to a random draw, the population size in a given urn i increases by one unit. Then, asymptotically, the population size increases linearly with time and proportionally to $(\deg(i) + 1)$.

In order to look at the evolution of our process described by the state vector $Z(t)$, we need to compute the probabilities $P(Z(t) = (j_1, \ldots, j_N))$ for each N-tuples $(j_1, \ldots, j_N)$, such that $j_i \geq 0$, for all $i = 1, \ldots, N$. To manipulate the N-tuples, we use classical conventions. We denote vectors with boldface characters, for example, $\mathbf{j} = (j_1, \ldots, j_n)$. We also expand the notation for the exponential in the classical way. That is, $\mathbf{s}^{\mathbf{j}}$ denotes the vector $(s_1^{j_1}, \ldots, s_N^{j_N})$ and similar notations for vector-valued functions.

A convenient way of pursuing the computations is to use the probability generating function given by

$$\begin{aligned} F(t, \mathbf{s}) &:= \sum_{\mathbf{j}=(j_1,\ldots,j_N)\geq 0} \mathbf{s}^{\mathbf{j}} P(Z(t) = j) \\ &= E\left(\mathbf{s}^{\mathbf{Z}(t)}\right) = E\left(s_1^{Z^1(t)} \cdot \ldots \cdot s_N^{Z_N(t)}\right), \end{aligned} \tag{2}$$

where $E(\cdot)$ is the expectation operator. The function $F(t, \mathbf{S})$ is called the probability generating function since it is possible to compute the probability function using it. In order to prove the existence of $F(t, \mathbf{s})$, we are now going to characterize the function by computing the Fokker-Planck equation [13, 14] (also known as Kolmogorov forward equation). Moreover, we are using this characterization to show that some results available in the literature are applicable to our model and lead to a time-based description of the interacting urn process.

In order to compute the probability generating function, we first compute $E(\mathbf{s}^{\mathbf{Z}(t+h)} \mid \mathbf{Z}(t))$, where $E(\cdot \mid \mathbf{Z}(t))$ is conditional expectation given that $\mathbf{Z}(t)$ is known. Using the description of the λ-exponential random time used in the proof of Lemma 1 shows that

$$\begin{aligned} &\mathbf{Z}(t+h) \\ &= \begin{cases} \mathbf{Z}(t) + \mathbf{n_k} & \forall k = 1, \ldots, N \text{ with probability } \lambda h + \mathcal{O}(h^2), \\ \mathbf{Z}(t) & \text{with probability } 1 - N\lambda h + \mathcal{O}(h^2), \end{cases} \end{aligned} \tag{3}$$

where the jth entry of the vector $\mathbf{n_k}$ is 0 or 1 to indicate that the jth artefact is a neighbor or not of k, and the vector $\mathbf{n_k}$ is a line of the adjacency matrix describing the network connections. We define that each artefact is its own neighbor. (This is a slight departure from the conventional notation.) In other words, adding the vector $\mathbf{n_k}$ to the state vector $\mathbf{Z}(t)$ corresponds to the situation where the artefact k proceeds to a random draw and adds a ball in its urn as well as in its neighbor's urns. Then, we get

$$E\left(\mathbf{s}^{\mathbf{Z}(t+h)} \mid \mathbf{Z}(t)\right) = \mathbf{s}^{\mathbf{Z}(t)} + \sum_{k=1}^{N} \left(\mathbf{s}^{\mathbf{Z}(t)+\mathbf{n_k}} - \mathbf{s}^{\mathbf{Z}(t)}\right)\lambda h + \mathcal{O}(h^2). \tag{4}$$

From (4), we obtain by computing the expectation on both side and passing to the limit $h \to 0$

$$\frac{dE\left(\mathbf{s}^{\mathbf{Z}(t)}\right)}{dt} = \sum_{k=1}^{N} E\left(\mathbf{s}^{\mathbf{Z}(t)}\right)(\mathbf{s}^{\mathbf{n_k}} - 1)\lambda. \tag{5}$$

Equation (5) is called the Fokker-Planck equation [13, 14] (also known as Kolmogorov forward equation). Solving this equation provides the complete knowledge of the probability that the state vector $\mathbf{Z}(t)$ is in a given state at time t. We are using this equation in order to derive another one which describes the evolution of the expectation of the number of balls in a given urn, that is, $E(Z_j(t)) = (\partial/\partial s_j)F(t, \mathbf{s})|_{\mathbf{s}=\mathbf{1}}$, for $j = 1, \ldots, N$ and with $\mathbf{1} = (1, \ldots, 1)^T$.

We first observe that $m_k(t) := (\partial/\partial s_k)E(\mathbf{s}^{\mathbf{Z}(t)})|_{\mathbf{s}=\mathbf{1}} = E(Z_k(t)\mathbf{s}^{\mathbf{Z}(t)-\mathbf{e_k}})|_{\mathbf{s}=\mathbf{1}}$. Secondly, by construction we have that asymptotically as $t \to \infty$ we have the approximation $Z_k(t) \approx t\lambda(\deg(k) + 1)$. We then have as the time t is large the approximation

$$\frac{\partial}{\partial s_k} E\left(\mathbf{s}^{\mathbf{Z}(t)}\right) \approx t\lambda(\deg(k) + 1)s_k^{-1} E\left(\mathbf{s}^{\mathbf{Z}(t)}\right). \tag{6}$$

By differentiating both sides of (5) with respect to s_j and using the approximation above, we obtain that for large t,

$$\frac{dE\left(\mathbf{s}^{\mathbf{Z}(t)}\right)}{dt}(t) \approx \sum_{k=1}^{N} \frac{1}{t(\deg(k) + 1)} (\mathbf{s}^{\mathbf{n_k}+\mathbf{e_k}} - s_k) \frac{\partial}{\partial s_k} E\left(\mathbf{s}^{\mathbf{Z}(t)}\right). \tag{7}$$

We then obtain a differential equation for $m_j(t)$ by differentiating both sides of this last equation with respect to $\partial/\partial s_j$ and evaluating the expression at $\mathbf{s} = \mathbf{1}$. This leads to

$$\frac{dm_j}{dt}(t) \approx \sum_{k=1}^{N} \frac{1}{t(\deg(k) + 1)} m_k(t)\delta_{j\sim k}, \tag{8}$$

where $\delta_{j\sim k}$ is 1 if j and k are neighboring artefacts and 0 else. In the limit $t \to \infty$, the limit $m(t)$ tends to the solution of (8) where the approx sign $\approx$ is replaced by an equality. This is due to the fact that the solution of (8) converges to a unique limit. In the following, we unashamedly proceed to the substitution. The term t is removed from the equation with the substitution $\tilde{m}(\log(t)) = m(t)$, from which we get

$$\frac{d\tilde{m}_j}{dt}(t) = \sum_{k=1}^{N} \frac{1}{\deg(k) + 1} \tilde{m}_k(t)\delta_{j\sim k}, \tag{9}$$

or written in matrix form and removing the tilde,

$$\begin{aligned} &\left(\frac{dm_1}{dt}(t), \ldots, \frac{dm_N}{dt}(t)\right) \\ &\quad = (m_1(t), \ldots, m_N(t))(I + D)^{-1}(I + A_d) \\ &\quad = (m_1(t), \ldots, m_N(t))A, \end{aligned} \tag{10}$$

where $A_d = (\delta_{i\sim j})_{i,j=1,\dots,N}$, I is the $N \times N$ identity matrix, and D is a diagonal matrix with $d_{ii} = 1/(\deg(i) + 1)$.

The FokkerPlanck equation (7) is one of a multitype branching process with constant rate ($\lambda^k(t) = \lambda^k$). From [12, 15], we know that

$$\lim_{t\to\infty} Z(t,\omega)e^{-\lambda_1 t} = W(\omega)u, \tag{11}$$

where $Z(t,\omega)$ is the state vector of our interacting urn process (1) in which we make explicit the underlying probability space by writing the sample point ω; $W(\omega)$ is a random variable and u is the left eigenvector of the matrix A corresponding to the largest eigenvalue λ_1. Hence, because of the exponential time change, the solution is going to converge like

$$\lim_{t\to\infty} \frac{Z(t,\omega)}{t^{-\lambda_1}} = W(\omega)u. \tag{12}$$

From (7) we know that given an initial urn content vector $(Z^1(0),\dots,Z^N(0))^T$, the expected population is given by

$$\left(m^1(0),\dots,m^N(0)\right)M(t) = \left(Z^1(0),\dots,Z^N(0)\right)\exp(At), \tag{13}$$

with the matrix A given by

$$a_{ij} = \frac{\delta_{ij}}{\deg(i)+1}, \quad i,j = 1,\dots,N. \tag{14}$$

This representation of the evolution of the vector $(m_1(t),\dots,m_N(t))$ is useful to prove a stronger result than (11) (or (12)). We state the result in the following proposition.

Proposition 1. *Let ξ be a right eigenvector of the matrix A and λ_ξ the corresponding eigenvalue. Then, the process defined by $Z(t)\cdot\xi\exp(-\lambda_\xi t)$ does converge as $t \to \infty$.*

Proof. The representation of the time evolution of the mean vector $(m_1(t),\dots,m_N(t))$ given by (13) shows that $m(t) = m(t_0)\exp(A(t-t_0))$, $\forall 0 \le t_0 \le t$. So, with $t_0 \le t$ fixed, we can write

$$\begin{aligned} m(t)\cdot\xi &= Z(t_0)\exp(A(t-t_0))\cdot\xi \\ &= Z(t_0)\cdot\xi\exp(\lambda_\xi(t-t_0)), \end{aligned} \tag{15}$$

and hence,

$$m(t)\cdot\xi\exp(-\lambda_\xi t) = Z(t_0)\cdot\xi\exp(-\lambda_\xi t_0). \tag{16}$$

This last equation shows that $Z(t)\cdot\xi\exp(-\lambda_\xi t)$ is a martingale. Indeed,

$$E(Z(t)\cdot\xi\exp(-\lambda_\xi t) \mid Z(t_0)) = Z(t_0)\cdot\xi\exp(-\lambda_\xi t_0). \tag{17}$$

The convergence follows from the classical convergence result for martingale. □

We also state the following corollary which will be useful in the next section.

Corollary 1. *We denote by $X_1, X_2, \dots$, the right eigenvectors of the matrix A and $\lambda_1, \lambda_2, \dots$ the corresponding eigenvalues. If we assume that the eigenvectors are independent, then we have the following decomposition:*

$$Z(t) = a_1(t)X_1\exp(\lambda_1 t) + a_2(t)\exp(\lambda_2 t) + \cdots, \tag{18}$$

where $a_i(t)$ is a random variable which converges as $t \to \infty$.

Proof. To compute the decomposition, we use the fact that by assumption the eigenvectors are orthogonal. To get the coefficient $a_i(t)$, we compute the scalar product of $Z(t)$ with the corresponding eigenvector X_i. The convergence follows from Proposition 1. □

The convergence rate of the process is determined by the second largest eigenvalue.

Proposition 2. *The eigenvalues $\lambda_1 > \lambda_2 \ge \cdots \ge \lambda_n$ of the matrix A are given by $\lambda_1 = 1 - \mu_i$ with $0 = \mu_1 < \mu_2 \le \cdots \le \mu_n$ being the eigenvalues of the Laplacian of the communication graph of the network.*

Proof. Let Ad be the adjacency matrix of the communication graph ($a_{ij} = 1_{i\sim j}$) and D the diagonal matrix with $d_{ii} = \deg(i)$. The matrix A is given by $A = (I+D)^{-1}(I + A_d)$. Algebraic manipulations show that if x, λ are right eigenvector and eigenvalue of A, then x, $1-\lambda$ are right eigenvector and eigenvalue of the Laplacian $D - A_d$ of the communication graph. The result follows since we assume that the communication graph is connected (then the eigenvalue 0 is simple) and by known properties of the spectrum of the Laplacian matrix. □

This proposition shows that the dominant eigenvalue of the matrix A is 1 and one can check that the corresponding left eigenvector is given by $(\deg(1)+1,\dots,\deg(N)+1)^T$.

Proposition 3. *Let x be the left eigenvector of A corresponding to the eigenvalue 1, then $u = (D+I)^{-1}x$ is a right generalized eigenvector of (A_d, D), that is, it satisfies $A_d u = Du$.*

Proof. The proof uses the same decomposition of the matrix A as in the previous proposition and proceeds by direct computations using the fact that the matrices $D + I$ and A_d are symmetric. □

Generalized eigenvectors are useful for spectral graph drawing. Actually, as discussed in [16], generalized eigenvectors corresponding to the generalized eigenvalues smaller than the dominant one provide coordinates for drawing the graph in the plane. Numerical evidence supports the fact that such generalized eigenvectors provide better coordinates than the eigenvectors of the Laplacian matrix. This proposition will support our application of the interacting urn process to clustering.

We now consider what does happen if we distinguish black and white balls in a same urn. Actually, the analysis previously presented carries on, the rate of split/die of the particles living in a given urn of different colors being asymptotically independent since $b_t^i + w_t^i \to t(\deg(i)+1)t$.

Theorem 1. *The scaled interacting urn process populations converge to a random vector which is proportional to the left eigenvector of the matrix A corresponding to the maximal eigenvalue* 1*; see* (12)*. The ratio of the black balls among the total population of a given urn, denoted X_t^i, converges to a mean value*

$$X_\infty^i = \frac{1}{\deg(i)+1}\sum_{j=1}^{N}\delta_{ij}X_\infty^j, \tag{19}$$

which is equivalent to

$$X_\infty^i = \frac{1}{\deg(i)}\sum_{j=1}^{N}1_{i\sim j}X_\infty^j. \tag{20}$$

Proof. The equivalence between the two expressions above follows from simple computation; we point out that the difference between the two sums is that in the first one we take into account the term X_∞^i while we do not in the second. The populations of black balls in the urn b_∞^i satisfy; see (12)

$$b_\infty^i = \sum_{j=1}^{n}\frac{\delta_{ij}}{\deg(j)+1}b_\infty^j. \tag{21}$$

Moreover, the total populations composing the urns satisfy

$$\frac{b_\infty^i + w_\infty^i}{\deg(i)+1} = \frac{b_\infty^j + w_\infty^j}{\deg(j)+1}, \tag{22}$$

because of the asymptotic limit as $t \to \infty$. Hence,

$$\begin{aligned} X_\infty^i &= \frac{b_\infty^i}{b_\infty^i + w_\infty^i} = \sum_j \frac{\delta_{ij}b_\infty^j}{(\deg(j)+1)(b_\infty^i + w_\infty^i)} \\ &= \sum_j \frac{\delta_{ij}b_\infty^j}{(\deg(i)+1)\left(b_\infty^j + w_\infty^j\right)} = \frac{1}{\deg(i)+1}\sum_j \delta_{ij}X_\infty^j. \end{aligned} \tag{23}$$

□

Broadly speaking, it is hard to get general results on the convergence rate of the interacting urn process because it is related to random processes (e.g., processes with reinforcement [1]). The process convergence rate depends on the value of the second largest eigenvalue of A. The difficulty of asserting general results is mainly because such results depend on the topology of the interactions, that is, the matrix A. However, the case where the topology of the graph of interactions is a complete graph is easy to understand, because the completeness implies identical urn content and the process is similar to the classical Pólya urn. This can motivate us to investigate the applications of the interacting urn process in clustered networks, since nodes that share many links are apt to develop similar urn content. Another *folk* result concerning processes similar to the interacting urn process is that the time for convergence to the definite value is usually very large and not numerically observable. However, some significant values may emerge quickly from the dynamic process. Moreover, notice that due to the probabilistic nature of the interacting urn process, fault tolerance is implied; there is an inherent recovery after the loss of a ball, say, due to communication interferences.

3. Applications

We now turn to describe two of the many possible applications of the interacting urn process in networks of tiny artifacts.

3.1. Cluster Formation.

3.1. Cluster Formation. Spectral graph drawing considers the entries of the generalized eigenvector of (A_d, D) corresponding to the second largest eigenvalue as $1d$ coordinate of the node as an efficient heuristic; see [16]. (By considering successive generalized eigenvectors of (A_d, D), one can also obtain multidimensional representation of the graph.) It is then natural to cluster nodes that are close in the $1d$ drawing of the graph.

3.1.1. The Analysis. We have shown that the population vector $(b_t^1, \ldots, b_t^N)/t$ converges, with the left eigenvector of the matrix A corresponding to the dominant eigenvalue 1, see equation 7, and that $b_t^i + w_t^i \to t(\deg(i)+1)$. The ratio $X_t^i = b_t^i/(b_t^i + w_t^i)$ is then converging to the generalized eigenvector of (A_d, D) corresponding to the dominant eigenvalue 1 by proposition 3, with a rate of convergence depending on the second largest eigenvalue. However, by direct computation, one can check that this generalized eigenvector is $(1, \ldots, 1)^T$, and by considering the difference $X_t^i - X_t^j$; we get an approximation of the component of the generalized eigenvalue of (A_d, D) corresponding to the second largest eigenvalue, which is related to the distance of the nodes in the $1d$ drawing of the graph.

The preceding analysis suggests running the interacting urn process and clustering neighbor nodes whose difference $X_t^i - X_t^j$ is below a threshold. In detail, we consider the vector of

$$\widetilde{Z}(t) = \left(\frac{b^1(t)}{b^1(t)+w^1(t)}, \cdots \frac{b^N(t)}{b^N(t)+w^N(t)}\right). \tag{24}$$

By (24), we have that $b^i(t) \to tu_1^i + t^{\lambda_2}u_1^i + \cdots +$ and $b^i(t) + w^i(t) \to (\deg(i)+1)t$. Therefore, we can write (25) (see Corollary 1),

$$\begin{aligned} \widetilde{Z}(t) &\longrightarrow a_1(w)X_1 + a_2(w)t^{\lambda_2-1}X_2 + \cdots \\ &= a_1(w)\begin{pmatrix}1\\ \vdots \\ 1\end{pmatrix} + a_1(w)t^{\lambda_2-1}X_2 + \cdots. \end{aligned} \tag{25}$$

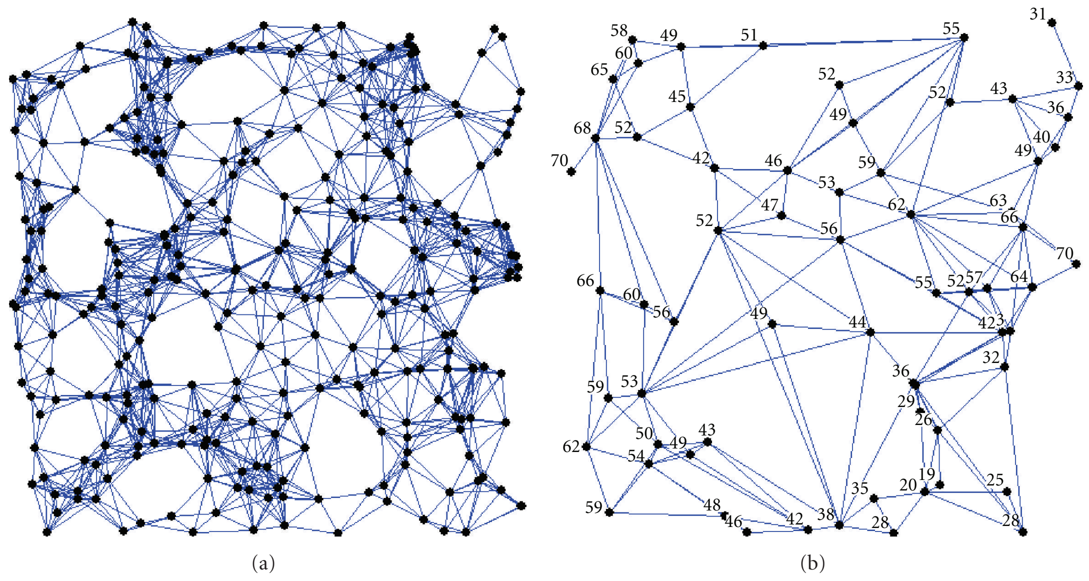


Figure 1: Clusters obtained with the interacting urn process. The left graph depicts the communication graph ($n = 300$ and $r = 0.12$). The right graph depicts the clusters with their assigned numerical values. For each cluster, we choose a representative node and connect representative nodes of other clusters, whenever there are neighboring nodes belonging to both clusters.

We note that $a_1(w)X_1$ refers to the generalized eigenvector and $\lambda_2 < 1$ but usually close to 1. In other words, we can write

$$\widetilde{Z}(t) = \begin{pmatrix} \dfrac{b^1(t)}{b^1(t)+w^1(t)} \\ \vdots \\ \dfrac{b^N(t)}{b^N(t)+w^N(t)} \end{pmatrix} \longrightarrow a_1(w)\begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} + a_1(w)t^{\lambda_2 - 1}X_2 + \cdots . \tag{26}$$

We conclude that the difference $b^i(t)/(b^i(t) + w^i(t)) - b^j(t)/(b^j(t)+w^j(t))$ is related to the differences $X_2^i - X_2^j$. This relation allows us to connect neighboring tiny artifacts for which the difference $X_2^i - X_2^j$ is small.

3.1.2. Numerical Results. A numerical study that validates our theoretical prediction is presented in Figure 1. In the experiment, the schedule of nodes interaction is generated uniformly at random. The time to stop the process was obtained while the clustered network stopped evolving visually, after about 20 draws per node; however, the global formation starts to emerge after 15 draws per node. In Figure 1, we observe that the algorithm behaves as expected, and a global cluster formation emerges.

3.2. Pseudogeometric Coordinates. In very large-scale networks, it is not possible to register the location of all nodes manually, or to equip all tiny artifacts with GPS units (such as [17]). Nevertheless, position-awareness is of great importance for many applications. For example, it is sometimes required or desirable to route without position information and to use *virtual coordinates* for georouting (see [18–20]). The algorithms for virtual coordinates assume that the underlying communication graph has a realization of a unit disk graph. The algorithms extract connectivity information of the communication graph and find a realization of virtual coordinates in the plane. For example, the realization can require that each edge has length at most 1 and that the distance between nonneighbored nodes is more than 1.

Rao et al. [18] describe an algorithm for computing virtual coordinates to enable geographic forwarding in a wireless ad hoc network. The mechanism in [18] is related to graph embedding. It is known that finding such a realization from connectivity information only (by using embedding of a unit disk graph) is NP hard (see [21]) and also hard to approximate (see [22]). Some applications do not require the realization of a unit disk graph. For example, georouting can be facilitated by *pseudogeometric coordinates* (see [23]). An algorithm for pseudogeometric coordinates selects several nodes, $a_1, \dots a_k$, as anchors, and assigns coordinate of each node, u, as its hop-distance to all anchors, that is, $(d(u, a_1), \dots d(u, a_k))$. It is required that the pseudogeometric coordinates uniquely identify each node. Moreover, the pseudogeometric coordinates must facilitate georouting. In this paper we follow an approach similar to the one in [19, 24] that uses information about the nodes' connectivity. The algorithm in [19] is an approximation algorithm that runs in polynomial time. In detail, the core of the mechanism in [18, Section 4.1] is described in (14); the values of the x_i, y_i coordinates of node p_i is the mean value of its neighbor coordinates

$$x_i = \frac{\sum_{P_k \in \text{neighbor\_set}(i)} x_k}{|\text{neighbor\_set}(i)|}; \qquad y_i = \frac{\sum_{P_k \in \text{neighbor\_set}(i)} y_k}{|\text{neighbor\_set}(i)|}. \tag{27}$$

After repeatedly letting each node p_i in the system to recalculate the values of the x_i, y_i coordinates, the eventual values are useful for greedy georouting (see [18, Section 4.1]). We suggest to replace the mechanism in [18, Section 4.1] with a simpler and cheaper mechanism of interacting urn processes.

3.2.1. The Process. We assume that merely a few anchor nodes have a registered position by some means, for example, using a GPS unit. We aim at letting all others nodes derive a coordinate system through connectivity information. Our approach is similar to that of Wattenhofer et al. [23], however, we are interested in using the interacting process as a heuristic that are applicable for *random geometric graphs.* (Random geometric graphs are constructed by dropping n points randomly uniformly into the unit square, $I = [0, 1]^2$, and adding edges to connect any two points distant at most r from each other (see [25] for the more general form of higher dimensions).) Consider the following process, in which the anchor nodes keep their coordinates unchanged during the entire process, and other nodes choose an initial position randomly. Then, repeatedly, the nodes transmit their positions, collect the coordinates of their neighbors, and assign themselves to the barycentric coordinates. The processes converge, producing pseudogeometric coordinates for the nodes.

We wish to emulate a similar process to the one described above, and let the interacting urn process estimate the mean of the barycentric neighboring values. Therefore, nodes maintain two urns, one for the x-axis and one for the y-axis, each urn composed of black and white balls. The position (x, y) of an anchor node corresponds to the color ratio of balls to be sent to their neighbors, that is, the urn with a constant number of x white balls and y black balls. Other nodes start with one black ball and one white ball in their urn. We let the interacting urn process run before using the content of both urns for producing the coordinates by taking the integer part of a factor of the color ratio in every urn. We identify nodes with the same integer coordinates as belonging to the same cluster, because of their similarity to the clustering application above.

3.2.2. Numerical Results. The obtained coordinates are presented in Figure 2. Similar to the clustering application, the time to stop the process was obtained while the pseudogeometric coordinate stopped evolving visually, after about 20 draws per node; however, the global formation starts to emerge after 15 drawings per node.

A statistical study is presented in Figure 3. Both charts consider a random geometric graph (Random geometric graphs are constructed by dropping n points randomly uniformly into the unit square, $I = [0, 1]^2$, and adding edges to connect any two points distant at most r from each other (see [25] for the more general form of higher dimensions).) of 50 nodes. Five anchor nodes are placed on the coordinate (0.0, 0.0), (0.0, 1.0), (0.5, 0.0), (0.5, 0.5), (0.5, 1.0), (1.0, 0.0), (1.0, 0.5), (1.0, 1.0). We study the statistics using Pearson's coefficient correlation (see [26]). An interesting question to find an answer for is how the number of hops from the anchor nodes to the other nodes statistically correlates to the Euclidian distance over the space of pseudogeometric coordinates. Moreover, we are interested in comparing with the alternative system of coordinates. Therefore, we present the statistical correlation of Euclidian distance over the space of the real coordinates (that where used to generate the communication graph) to the number of hops from them to the anchor nodes.

In more detail, for every two nodes, u and v, let us define $\text{dis}_{\text{pseudo}}(u, v)$ and $\text{dis}_{\text{real}}(u, v)$ as the Euclidian distances $\sqrt{(u.x_{\text{pseudo}} - v.x_{\text{pseudo}})^2 + (u.y_{\text{pseudo}} - v.y_{\text{pseudo}})^2}$, and, respectively, $\sqrt{(u.x_{\text{real}} - v.x_{\text{real}})^2 + (u.y_{\text{real}} - v.y_{\text{real}})^2}$, where $(u.x_{\text{pseudo}}, v.y_{\text{pseudo}})$ and $(u.x_{\text{real}}, v.y_{\text{real}})$ are the pseudogeometrical, and, respectively, real coordinates of nodes u. Moreover, $\text{hop}(u, v)$ is the number of hops on the shortest path between nodes u and v. The statistical correlation between $\text{dis}_{\text{pseudo}}(u, v)$ and $\text{hop}(u, v)$ is presented in the left chart of Figure 3, where u is the anchor node at (0.5, 0.5) and v is any of the 50 nodes on the random-geometric graph that are not anchored. We see that after 5 rounds, the value of the correlation starts to oscillate around the average value of 0.85 and with small (standard) deviation (of less than 0.01). The left chart of Figure 3 also presents the alternative correlation. Namely, 0.75 is the value of the statistical correlation between $\text{dis}_{\text{real}}(u, v)$ and $\text{hop}(u, v)$, where u is the anchor node at (0.5, 0.5) and v is any of the 50 nodes that are not anchored.

Georouting facilitates the communication between any pair of nodes (that are not necessarily one of the anchor nodes). Therefore, we consider the statistical correlation between $\text{dis}_{\text{pseudo}}(u, v)$ and $\text{hop}(u, v)$, where u and v are any two nodes on the random-geometric graph that are not anchored. The values of the statistical correlation are presented in the right chart of Figure 3 as a function of the number of rounds. Moreover, the right chart also depicts the linear regression of these values.

The correlation values that are presented in Figure 3 suggest that (over time), the pseudogeometric coordinates can facilitate georouting better than real coordinates, because of their stronger statistical coalition to the hop distance.

4. The Implementation

We present a self-stabilizing implementation of the urn process. The pseudocode of the implementation is given in Algorithm 1. Before we explain the design, we list the key assumptions used in the analysis of interacting urn processes. For instance, it assumes that time is continuous. However, in practice, clock mechanisms are discrete. Fortunately, when considering a stochastic process evolving in continuous

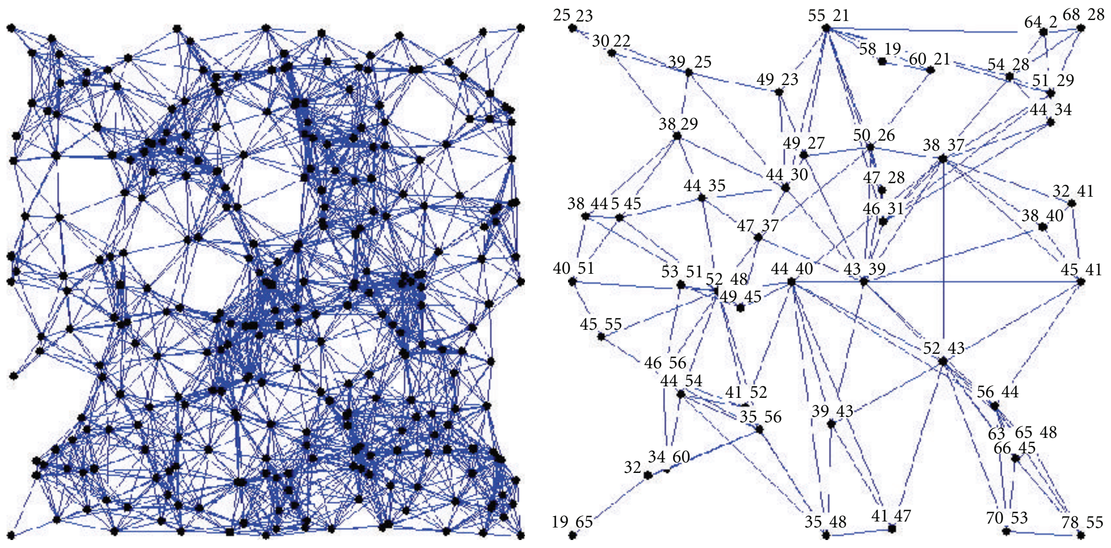
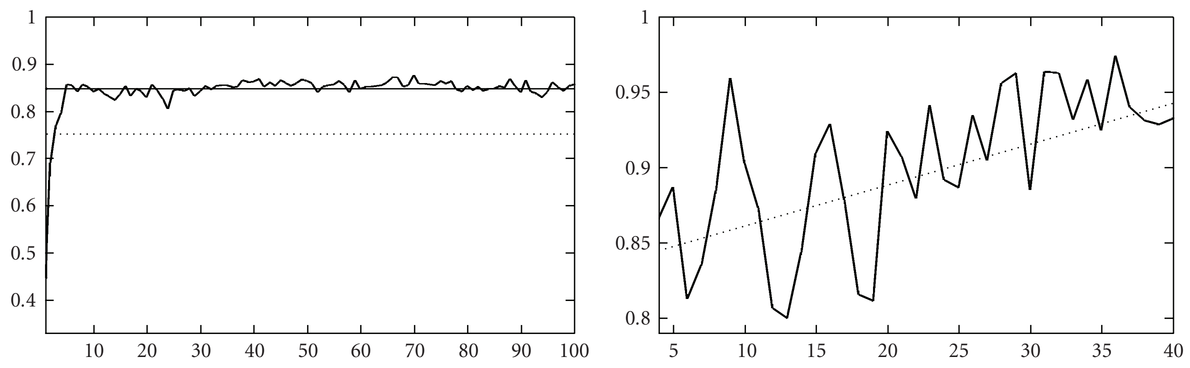


Figure 2: Pseudogeometric coordinates obtained with two interacting urn processes. On the left graph, we see the communication graph ($n = 300$ and $r = 0.12$) with 8 anchor nodes at $(0.0, 0.0)$, $(0.0, 1.0)$, $(0.5, 0.0)$, $(0.5, 0.5)$, $(0.5, 1.0)$, $(1.0, 0.0)$, $(1.0, 0.5)$, $(1.0, 1.0)$. For the sake of presentation simplicity, the pseudogeometric coordinates are depicted for a representative member of their clusters. The numbers are the colors' ratio multiplied by 100. See Figure 1 for the clusters' description.



(a) Communicating with the anchor nodes (b) Communicating among nodes that are not anchored

Figure 3: Pearson's coefficient correlation between $(u.x_{\text{real}}, v.y_{\text{real}})$ and $\text{hop}(u, v)$ (cf. [27, 28]). The x's axes describe the number of communication rounds and the y's axes describe the coefficients.

time, it is always possible to conduct a discrete process by considering merely the produced successive events and ignore any reference to continuous time. The transformation from continuous to discrete stochastic model is known as *discrete skeleton* (see [29]).

4.1. Serializable of Algorithm Steps. Another assumption that we make in our analysis is about the instantiation of the algorithm steps. Namely, the analysis assumes that the algorithm steps are serializable, that is, it takes no time to draw a ball, announce it, and let the drawing tiny artifact and the neighboring tiny artifacts update their urns.

In real distributed systems, these assumptions do not always hold and concurrent algorithm steps can be nonserializable. Therefore, we require that at any time and any neighborhood, there is at most one tiny artifact that takes an algorithm step. In other words, no two nodes that have a path of less than three hops may concurrently take the algorithm steps.

In order to provide serializability of the algorithm steps, we borrow existing self-stabilizing infrastructure. Herman and Tixeuil [30] present a self-stabilizing algorithm for accessing a shared media, such as the communication environment of wireless tiny artifacts. The algorithm assures that starting for an arbitrary configuration of the system, eventually every node has a unique time slot for broadcasting. Namely, no node broadcasts concurrently with a neighboring node. The property of time slot uniqueness facilitates serializable steps of the algorithm that is presented in Algorithm 1. In other words, when the broadcasting time slot of a particular tiny artifact arrives, this tiny artifact can

```
   Const
2    T: the parameter of the floating output

4  Types
     colors: { black, white }
6    timeouts: {URN, FLOAT}

8  Variables
     urn: multiset of colors, initially ⟨black, white ⟩
10   output: the output value of the algorithm

12 External functions
     send() / receive(): data link layer interface
14   select(): a uniform selection
     set-timer(timeouts, time) / timer_expired(timeouts): timer interface
16
   Macro
18   time2wait(): choose time to wait using λ-exponential distribution

20 Upon receive(⟨ball ⟩)
     urn ← urn ∪ { ball }
22
   Upon timer_expired(type)
24   if type = URN then
       let ball = select(urn)
26     send(⟨ball ⟩)
       urn ← urn ∪ { ball }
28     set-timer(URN, time2wait())
     else
30     output ← urn
       urn ← ⟨black, white ⟩
32     set-timer(FLOAT, T)
```

Algorithm 1: The interacting urn algorithm.

draw a ball (see line 25), announce it using a broadcast (that eventually does not collide; see line 26), and let the neighboring tiny artifacts update their urns; see line 21 (as well as updating its own urn; see line 27). Hence, eventually the algorithm steps are serializable.

4.2. Self-Stabilization. Self-stabilizing systems [7, 8] can recover after the occurrence of transient faults. These systems are designed to automatically regain their consistency from any starting state. The arbitrary state may be the result of violating the assumptions about the system settings and assist with dealing with the self-organization requirements of ad hoc networks. The correctness of a self-stabilizing system is demonstrated by considering every sequence of actions that follows the last transient fault and is, therefore, proved assuming an arbitrary starting state of the automaton.

The property of serializability is guaranteed to be archived eventually. We note that since this property does not always hold, it implies, for example, that while the network of tiny artifacts is deployed, the property of serializability can be violated.

The technique of *floating output* (see [8]) is a way to convert a nonstabilizing algorithm that computes a fixed output into a self-stabilizing algorithm. We use the technique of floating output in order to overcome the scenarios in which the property of serializability is violated during the deployment of the network.

The algorithm executes the interacting urn process for a sufficiently long period, T, that allows the interacting urn processes to produce the correct output (assuming that the property of serializable steps is never violated). We name by *parameter of the floating output* the constant T.

We assume that the system has access to a self-stabilizing clock synchronization mechanism (such as [31, 32]), that facilitates the agreement on a particular time slot once in every T time slot. The algorithm makes sure that in that time slot (1) every tiny artifact stores the values of all urns in $output$ (see line 30), (2) no tiny artifact draws a ball from its urn (see line 31), (3) every tiny artifact restarts its state (by assigning the urn with the initial value of one black ball and one white ball; see line 32). We note that the values in $output$ are used for calculating the output (e.g., for producing the pseudogeometric coordinates).

4.2.1. Correctness. Demonstrating that the algorithm presented in Algorithm 1 is self-stabilizing is quite simple and is followed by the conventional arguments of the technique of floating output (see [8]).

We consider the period that is after the network was deployed and the media access algorithm has stabilized to work correctly. It is required to demonstrate that within a period of T, the variable $output$ contains the correct output. Let us consider the first timeout of the type FLOAT. Within a period of T after that timeout, the interacting urn processes produce the correct output (by the definition of T). Moreover, at the end of that period, all tiny artifacts assign the correct output to variable $output$ (see line 30). By similar arguments, all subsequent periods produce the correct output as well. Thus, the variable $output$ shows the correct output in all of the subsequent periods.

5. Conclusions

We are interested in simplifying the design of tiny artifacts and bridging the gap between these future networks and existing ones. Existing implementations, say, for sensor networks, often use protocols that assume traditional system settings that require resources that tiny artifacts do not have. Alternatively, when the designers do not assume traditional system settings, they turn to improving performance and reducing resource consumption by using probabilistic algorithms. However, designers that do not consider implementation explicitly do not specify the exact computational power required for each node. In some cases, the implementation requires storing nontrivial quantities of data.

This work presents a self-stabilizing building block that can facilitate infrastructure for tiny artifacts, such as reasonable clustering and efficient georouting. Our analytical and numerical results show that global formations appear rapidly (with the use of small number of transmissions and few bits at a time).

Acknowledgments

This work would not have been possible without the contribution of Paul G. Spirakis in many helpful discussions, ideas and analysis. Many thanks are due to Edna Oxman for improving the presentation. The authors are thankful for the code provided by the students of the course DATX01-15 (Chalmers University of Technology), 2008. This work has been partially supported by the Swiss SER Contract no. C05.0030, and by the ICT Programme of the European Union under contract number FP7-215270 (FRONTS). An extended abstract of this paper appeared in [33].

References

[1] R. Pemantle, "A survey of random processes with reinforcement," *Probability Surveys*, vol. 4, pp. 1–79, 2007.

[2] N. L. Johnson and S. Kotz, *Urn Models and Their Applications: An Approach to Modern Discrete Probability Theory*, John Wiley & Sons, New York, NY, USA, 1977.

[3] B. Arthur, Y. Ermoliev, and Y. Kaniovski, "Path dependent processes and the emergence of macrostructure," in *Increasing Returns and Path Dependence in the Economy*, B. Arthur, Ed., chapter 3, pp. 33–48, The University of Michigan Press, Ann Arbor, Mich, USA, 1994.

[4] N. Tishby and N. Slonim, "Data clustering by Markovian relaxation and the information bottleneck method," in *Advances in Neural Information Processing Systems (NIPS '00)*, T. K. Leen, T. G. Dietterich, and V. Tresp, Eds., pp. 640–646, MIT Press, Denver, Colo, USA, 2000.

[5] D. Angluin, J. Aspnes, Z. Diamadi, M. J. Fischer, and R. Peralta, "Urn automata," Tech. Rep. YALEU/DCS/TR1280, Department of Computer Science, Yale University, New Haven, Conn, USA, November 2003.

[6] N. L. Johnson and S. Kotz, *Urn Models and Their Applications: An Approach to Modern Discrete Probability Theory*, John Wiley & Sons, New York, NY, USA, 1977.

[7] E. W. Dijkstra, "Self-stabilizing systems in spite of distributed control," *Communications of the ACM*, vol. 17, no. 11, pp. 643–644, 1974.

[8] S. Dolev, *Self-Stabilization*, MIT Press, Cambridge, Mass, USA, 2000.

[9] F. Heylighen and C. Gershenson, "The Meaning of Self-organization in Computing," *IEEE Intelligent Systems*, vol. 18, no. 4, pp. 72–75.

[10] C. Gershenson and F. Heylighen, "Protocol requirements for selforganizing artifacts: towards an ambient intelligence," in *Proceedings of International Conference on Complex Systems (ICCS '04)*, pp. 497–503, ACM Press, Boston, Mass, USA, May 2004.

[11] C. Gershenson and F Heylighen, "When can we call a system self-organizing?" in *Proceedings of the 7th European Conference on Artificial Life (ECAL '03)*, W. Banzhaf, T. Christaller, P. Dittrich, J. T. Kim, and J. Ziegler, Eds., vol. 2801 of *Lecture Notes in Computer Science*, pp. 606–614, Springer, Dortmund, Germany, 2003.

[12] P. E. Ney and K. B. Athreya, *Branching Processes*, Courier Dover Publications, Mineola, NY, USA, 2004.

[13] A. Papoulis, S. U. Pillai, A. Papoulis, and S. U. Pillai, *Probability, Random Variables, and Stochastic Processes*, McGraw-Hill, New York, NY, USA, 1965.

[14] L. P. Kadanoff, *Statistical Physics: Statics, Dynamics and Renormalization*, World Scientific, Singapore, 2000.

[15] K. B. Athreya, "Some results on multitype continuous time Markov branching processes," *The Annals of Mathematical Statistics*, vol. 39, no. 2, pp. 347–357, 1968.

[16] Y. Koren, "On spectral graph drawing," in *Proceedings of the 9th Annual International Conference on Computing and Combinatorics (COCOON '03)*, T. Warnow and B. Zhu, Eds., vol. 2697 of *Lecture Notes in Computer Science*, pp. 496–508, Springer, Big Sky, MT, USA, July 2003.

[17] B. Hofmann-Wellenhof, H. Lichtenegger, and J. Collins, *GPS Theory and Practice*, Springer, New York, NY, USA, 2001.

[18] A Rao, C. H. Papadimitriou, S. Shenker, and I. Stoica, "Geographic routing without location information," in *Proceedings of the 9th Annual International Conference on Mobile Computing and Networking (MOBICOM '03)*, D. B. Johnson, A. D. Joseph, and N. H. Vaidya, Eds., pp. 96–108, ACM, San Diego, Calif, USA, 2003.

[19] T. Moscibroda, R. O'Dell, M. Wattenhofer, and R. Wattenhofer, "Virtual coordinates for ad hoc and sensor networks," in *Proceedings of the DIALM-POMC Joint Workshop on Foundations of Mobile Computing*, S. Basagni and C. A. Phillips, Eds., pp. 8–16, Philadelphia, Pa, USA, October 2004.

[20] B. Leong, B. Liskov, and R. Morris, "Greedy virtual coordinates for geographic routing," in *Proceedings of the IEEE*

International Conference on Network Protocols (ICNP '07), pp. 71–80, IEEE, Beijing, China, October 2007.

[21] H. Breu and D. G. Kirkpatrick, "Unit disk graph recognition is NP-hard," *Computational Geometry: Theory and Applications*, vol. 9, no. 1-2, pp. 3–24, 1998.

[22] F. Kuhn, T. Moscibroda, and R. Wattenhofer, "Unit disk graph approximation," in *Proceedings of the Joint Workshop on Foundations of Mobile Computing*, S. Basagni and C. A. Phillips, Eds., pp. 17–23, Philadelphia, Pa, USA, 2004.

[23] M. Wattenhofer, R. Wattenhofer, and P. Widmayer, "Geometric routing without geometry," in *Proceedings of the 12th Colloquia on Structural Information and Communication Complexity (SIROCCO '05)*, A. Pelc and M. Raynal, Eds., vol. 3499 of *Lecture Notes in Computer Science*, pp. 307–322, Springer, 2005.

[24] R. Bischoff and R. Wattenhofer, "Analyzing connectivity-based multi-hop ad-hoc positioning," in *Proceedings of the 2nd IEEE International Conference on Pervasive Computing and Communications (PerCom '04)*, pp. 165–176, IEEE Computer Society, 2004.

[25] M. Penrose, *Random Geometric Graphs*, Oxford University Press, Oxford, UK, 2003.

[26] D. S. Moore, *The Basic Practice of Statistics*, W. H. Freeman, New York, NY, USA, 4th edition, 2006.

[27] J. L. Rodgers and W. A. Nicewander, "Thirteen ways to look at the correlation coefficient," *The American Statistician*, vol. 42, no. 1, pp. 59–66, 1988.

[28] J. L. Rodgers, W. A. Nicewander, and L. Toothaker, "Linearly independent, orthogonal, and uncorrelated variables," *The American Statistician*, vol. 38, no. 2, pp. 133–134, 1984.

[29] P. Guttorp, *Stochastic Modeling of Scientific Data*, Chapman & Hall/CRC, London, UK, 1995.

[30] T. Herman and S. Tixeuil, "A distributed TDMA slot assignment algorithm for wireless sensor networks," in *Proceedings of the 1st International Workshop on Algorithmic Aspects of Wireless Sensor Networks (ALGOSENSORS '04)*, vol. 3121 of *Lecture Notes in Computer Science*, pp. 45–58, Springer, Turku, Finland, July 2004.

[31] T. Herman and C. Zhang, "Best paper: stabilizing clock synchronization for wireless sensor networks," in *Proceedings of the 8th International Symposium on Stabilization, Safety, and Security of Distributed Systems (SSS '06)*, A. K. Datta and M. Gradinariu, Eds., vol. 4280 of *Lecture Notes in Computer Science*, pp. 335–349, Springer, Dallas, Tex, USA, November 2006.

[32] J.-H. Hoepman, A. Larsson, E. M. Schiller, and P. Tsigas, "Secure and self-stabilizing clock synchronization in sensor networks," in *Proceedings of the 9th International Conference on Stabilization, Safety, and Security of Distributed Systems (SSS '07)*, T. Masuzawa and S. Tixeuil, Eds., vol. 4838 of *Lecture Notes in Computer Science*, pp. 340–356, Springer, 2007.

[33] P. Leone and E. M. Schiller, "Interacting urns processes: for clustering of large-scale networks of tiny artifacts," in *Proceedings of the ACM Symposium on Applied Computing (SAC '08)*, R. L. Wainwright and H. Haddad, Eds., pp. 2046–2051, ACM, Fortaleza, Brazil, 2008.