Serdica J. Computing **8** (2014), No 4, 363–388

Serdica
Journal of Computing

Bulgarian Academy of Sciences
Institute of Mathematics and Informatics

AN ADAPTATION OF THE HOSHEN-KOPELMAN CLUSTER COUNTING ALGORITHM FOR HONEYCOMB NETWORKS

Hristina Popova

Abstract. We develop a simplified implementation of the Hoshen-Kopelman cluster counting algorithm adapted for honeycomb networks. In our implementation of the algorithm we assume that all nodes in the network are occupied and links between nodes can be intact or broken. The algorithm counts how many clusters there are in the network and determines which nodes belong to each cluster. The network information is stored into two sets of data. The first one is related to the connectivity of the nodes and the second one to the state of links. The algorithm finds all clusters in only one scan across the network and thereafter cluster relabeling operates on a vector whose size is much smaller than the size of the network. Counting the number of clusters of each size, the algorithm determines the cluster size probability distribution from which the mean cluster size parameter can be estimated. Although our implementation of the Hoshen-Kopelman algorithm works only for networks with a honeycomb (hexagonal) structure, it can be easily changed to be applied for networks with arbitrary connectivity between the nodes (triangular, square, etc.). The proposed adaptation of the Hoshen-Kopelman cluster counting algorithm is applied to studying the thermal degradation of a graphene-like honeycomb membrane by means of Molecular Dynamics simulation with a Langevin thermostat.

ACM Computing Classification System (1998): F.2.2, I.5.3.
Key words: cluster counting algorithm, honeycomb network, molecular dynamics simulation.

1. Introduction. The performance of algorithms to find and label clusters is of great importance for the simulation of percolation phenomena. The percolation problem was first posed in 1957 by Broadbent and Hammersley [1] and the first effective simulation technique that could simulate this problem in an efficient way was proposed in 1976 by Hoshen and Kopelman [2]. The introduction of the Hoshen-Kopelman algorithm was an important breakthrough in the analysis of cluster size statistics in percolation theory [3, 4]. Only after the introduction of this algorithm, did Monte-Carlo simulations of very large lattices become possible [5, 6, 7]. The algorithm's single and sequential pass through the lattice linearizes the time and memory space requirement as a function of the lattice size [8]. Before the Hoshen-Kopelman algorithm it was believed that the computational efforts grow faster than the squared number of particles subject to clustering. The Hoshen-Kopelman algorithm proved that this relationship can be linear (the algorithm finds all clusters in only one scan across the lattice). That was the real breakthrough, because very often, especially for percolation models, the number of particles in the system might be more than 10^6. Moreover, the Hoshen-Kopelman algorithm solved the serious problem of lack of computer memory for very large percolation systems. When the algorithm was developed, saving memory and computation time was a crucial issue for getting results in reasonable computing time. Nowadays such intelligent algorithms are still very helpful, e.g. for investigating very large systems.

Although the algorithm was initially applied in statistical physics, nowadays it is applied in many diverse fields [9, 10, 11, 12, 13, 14]. The concept of percolation has been useful in describing a variety of physical, chemical, and biological phenomena [15, 16, 17]. Among the typical applications of the percolation theory one may also find material science [18, 19, 20], immunology [21, 22, 23, 24], or forest fires problems [25, 26, 27] and studies of liquids moving in porous media [28, 29, 30], etc. Generally speaking, the percolation theory deals with statistical properties of the clusters of occupied nodes (site percolation) or occupied edges (bond percolation) for a given graph, network or regular lattice. So, two distinct types of percolation processes are recognized: site percolation and bond percolation. Permeation of fluids through porous media [31, 32, 33] and gel formation by polymers via cross-linking [34] can be explained in terms of the bond percolation theory [15], whereas crystal phenomena, such as spontaneous magnetization of dilute ferromagnets [35], diffusion in binary alloys [36, 37], and exciton percolation in molecular crystals [38, 39], are described within the framework of site percolation. Site and bond percolation processes have both been suggested for electrical conductivity models of disordered materials [40, 41, 42, 43]. But the algorithm is not restricted to pure site or pure bond lattices; it has been extended to lattices that consist of sites and bonds [44, 45, 46, 47]. Furthermore, in 1997 the Hoshen-Kopelman algorithm was extended [48] to determine information not only on the cluster size but also on the structure of the clusters, such as the radius of gyration, internal perimeter, or spatial moments. This enhanced Hoshen-Kopelman algorithm was applied to large images [49], and used to calculate cluster properties and entropy in percolation models [50]. Different approaches were proposed to parallelize the Hoshen-Kopelman algorithm [51, 52, 53, 54, 55].

Most adaptations and implementations of the Hoshen-Kopelman algorithm were for lattice environments (discrete systems). Only a few studies were devoted to discussing the Hoshen-Kopelman algorithm implementation in non-lattice environments (continuous systems). In such systems, the positions of the sites are arbitrary, and not restricted to the discrete points of a regular lattice. Gawlinski and Stanley [56] were the first to adapt the Hoshen-Kopelman algorithm to handle continuum percolation by overlaying an imaginary covering mesh onto a square area. Their adaptation was implemented in other continuum percolation models on discs and spheres [57, 58]. Non-lattice environments exist not only in the percolation theory of disordered discs and spheres, but also in the networks of many interacting units that are observed in complex systems. Researchers are only now beginning to unravel the structure and dynamics of such complex networks [59, 60].

In the paper by Al-Futaisi and Patzek [61] in 2003, the Hoshen-Kopelman algorithm for cluster labeling was extended to non-lattice environments where network elements (sites or bonds) are placed at random points in space. This extension of the Hoshen-Kopelman algorithm is not restricted to a non-lattice environment, and can be applied to lattices in two, three or higher dimensions. It can also be applied to networks consisting of sites (nodes), bonds, or both, and each site can have a different number of connecting bonds. Then clusters of sites, bonds, or sites-and-bonds in such a complex arbitrary network can be labeled with this implementation of the Hoshen-Kopelman algorithm. Using this extended algorithm, Al-Futaisi and Patzek studied the sizes of oil clusters trapped in a disordered pore network extracted from a 3D micro-focused X-ray CT image of Bentheimer sandstone (a real porous rock). The extension of the Hoshen-Kopelman algorithm should be useful in the studies of percolation in continuum systems and in the detection of fluid clusters in more realistic networks extracted from complex rocks. Following Al-Futaisi and Patzek [61], we develop our simplified implementation of the cluster counting algorithm adapted for networks with honeycomb structure. Thereafter we studied the degradation process [62, 63] of a two-dimensional membrane model with such honeycomb structure, as shown in Fig. 1.

The paper is organized as follows. Next, in Section 2, we describe the cluster counting problem. Then, in Section 3, we present our implementation of the Hoshen-Kopelman cluster counting algorithm for honeycomb networks. After that, in Section 4, we apply our cluster counting procedure to a graphene-like honeycomb membrane in a process of thermal degradation. Finally, we end the paper with a brief summary in Section 5. A detailed description of our implementation of the Hoshen-Kopelman cluster counting algorithm in the C programming language is given in the Appendix.

2. Cluster counting problem. Let us imagine that we have a large number of particles spread somehow in space and subject to clustering. Particle here means any geometrical object which could be connected under certain rules with some of its neighbors. Such objects could be atoms, monomers, polymers, sand grains, ... telephones, computers, computer networks, ... stars, galaxies. Each set of connected particles forms a connectivity (cluster). For each configuration, we will need to extract some statistics for the distribution of the clusters-their size, form, fractal dimension and so on. Then a statistics for an ensemble of configurations could be made.

So, the cluster counting could be thought as the following task:

1. Create a (computer) model structure of the particles for which the spread of connectivity will be studied. In some cases that could be a digitized image of natural objects, e.g. a colony of bacteria.

2. Decide, for each pair of particles, if they are adjacent (bonded) or not.

3. Identify the clusters of connected particles.

4. Make statistics of certain properties of the clusters. Store the statistics for further use.

5. Repeat items 1–4 (or 2–4) enough times in order to have reliable counts for the statistics made within each single realization; or, follow the evolution of the cluster statistics with time, if the realizations are not independent.

The difficulties are concentrated mainly in item 3 of the list above and the present paper focuses on it.

3. Implementation of the cluster counting algorithm. We provide a simplified implementation of the Hoshen-Kopelman cluster counting algorithm for honeycomb networks in non-lattice environments. We develop this algorithm to study the thermal decomposition of a membrane with hexagonal shape and honeycomb structure [62] (see Fig. 1). In such a honeycomb network each node has at most three connecting bonds with the nearest-neighboring nodes, except for the peripheral nodes on the edges of our hexagonal network which have at most two connections with neighboring nodes. Although our implementation of the Hoshen-Kopelman algorithm is simplified and works only for networks with a honeycomb structure, it can be easily changed to be applied for networks with arbitrary connectivity between the nodes (triangular, square, and so on structure).

In our implementation of the Hoshen-Kopelman algorithm we assume that all nodes and all links in the network are occupied, but some links can be broken when the network is subject to the process of thermal degradation. The algorithm counts how many clusters there are in the given network and determines which nodes belong to each cluster. The network information is stored in two arrays. The first one is related to the connectivity of the nodes and the second one to the state of links (intact or broken link). The connectivity of the nodes in the network is described through the **Bond_Neighbour** array. In this array, we define the neighboring nodes that are directly connected to each node. The size of this array is the total number N of nodes in the network by the maximum connectivity or the coordination number (which is 3 in a honeycomb network). Therefore for the nodes that have a coordination number less than the maximum we define their appropriate neighboring nodes and assign minus one (-1) to the remaining array elements (such are the peripheral nodes). The other array **bond_rupture** describes the state of each bond. If the bond is intact, we assign zero (0), otherwise when the bond is broken, we assign one (1).

With these data arrays in hand, we are ready to describe our implementation of the Hoshen-Kopelman algorithm. The description of the algorithm is also translated into the C programming language in the Appendix. In what follows, our implementation of the Hoshen-Kopelman cluster counting algorithm is described in six steps:

1. Read the network data structures-the input (global) arrays **Bond_Neighbour** and **bond_rupture**.
2. Initialize the output arrays **NodeL** which stores a cluster label for each node.
3. Create an empty output array **NodeLP** which holds information about the cluster labels.

4. Scan the network nodes in their consecutive order. At each node, i, as we traverse the network, elements in **NodeL** and **NodeLP** are changed according to the following conditions:
   (4.1) If the node does not have any links together with its adjacent (neighboring) nodes, then start a new cluster, and record this new cluster in **NodeL** and **NodeLP**.
   (4.2) If the node has at least one link, then we define two possibilities:
      (i) If none of the neighboring nodes is labeled, then start a new cluster, and update **NodeL** and **NodeLP**, accordingly.
      (ii) If there exists a labeled neighbor and its link also exists, then set **NodeL** of the node and **NodeLP** of the neighboring nodes equal to the minimum of **NodeLP[NodeL[j]]** of the neighboring nodes j.
5. After the scan in Step 4 is completed, operate only on the array **NodeLP**, and renumber labels in **NodeLP** to be sequential.
6. Apply the corrected labels in **NodeLP** to the array **NodeL**.

Just as for the classic (original) Hoshen-Kopelman algorithm on lattices and for the extended Hoshen-Kopelman algorithm on continuum systems, our implementation of the algorithm finds all clusters in a single pass through the nodes of the network and cluster relabeling operates on a vector whose size is much smaller than the size of the network. Finally, we record the total number of clusters found in the network, and perform cluster statistics. To determine the cluster size distribution (or probability distribution of fragment sizes), we have to count the number of clusters of each size (and to store these data in the array **NC**, which has to be normalized at the end).

4. Application: a thermal degradation of a graphene-like honeycomb membrane. The proposed adaptation of the Hoshen-Kopelman cluster counting algorithm is applied to study the thermal degradation of a graphene-like honeycomb membrane by means of Molecular Dynamics simulation with a Langevin thermostat.

A. Thermal degradation problem. Thermal degradation and stabilization of polymer systems has been a long-standing focus of research from both practical and fundamental viewpoints [64]. Plastic waste disposal has grown rapidly to an ecological menace prompting researchers to investigate plastic recycling by degradation as an alternative [65]. On the other hand, degradation of polymers and other high molecular weight materials in different environments is usually a major limiting factor in their application. Thermal degradation (or, *thermolysis*) plays a decisive role in the design of flame-resistant polyethylene and other plastic materials [66]. Another interesting aspect for applications includes reversible polymer networks [67, 68], and most notably, graphene, as a "material of the future" that shows unusual thermomechanical properties [69]. Recently, with the rapidly growing perspective of exploiting bio-polymers as functional materials [70, 71] the stability of such materials has become an issue of primary concern [72, 73] as, e.g., that of double-stranded polymer decomposition [74]. Most theoretical and computational investigations have been focused on degradation of polymer chains with one-dimensional (1D) topological connectivity [75, 76, 77, 78, 79, 80, 81]. Understanding the interplay between elastic and fracture properties is even more challenging and important in the case of two-dimensional (2D) polymerized networks (elastic-brittle sheets). A prominent example of biological microstructure is *spectrin*, the red blood cell membrane skeleton, which reinforces the cytoplasmic face of the membrane. In erythrocytes, the membrane skeleton enables it to undergo large extensional deformations while maintaining the structural integrity of the membrane. A number of studies, based on continuum-level [82], percolation-level [83, 84, 85], or molecular-level [86, 87] considerations of the mechanical breakdown of this network, modeled as a triangular lattice of spectrin tetramers, have been reported so far. Another example concerns the thermal stability of isolated graphene nanoflakes [88]. Some studies consider the problem of thermal decomposition of gels [89, 90], epoxy resins [91, 92] and other three-dimensional (3D) networks, studied both experimentally [89, 90, 91, 92], and by means of simulations [93] as in the case of Poly-dimethylsiloxane (PDMS). In most of these cases, however, mainly a stability analysis is carried out whereas still little is known regarding the collective mechanism of degradation, the dependence of rupture time on system size, as well as the decomposition kinetics, especially as far as 2D polymer network sheets are concerned. Therefore, in a recent work [62] using Molecular Dynamics (MD) simulation we extended the investigations to the case of 2D polymer network sheets, embedded in 3D-space, and studied as a generic example the thermal degradation of a suspended membrane with honeycomb orientation, similar to that of graphene.

B. Model. We study a coarse-grained model of honeycomb membrane embedded in three-dimensional space. The membrane consists of N spherical particles (beads, monomers) of diameter σ connected in a honeycomb lattice structure whereby each monomer is bonded with three nearest neighbors except for the monomers on the membrane edges which have only two bonds (see Fig. 1). The total number of monomers N in such a membrane is $N = 6L^2$ where by L we denote the number of monomers (or hexagonal cells) on the edge of the membrane (i.e., L characterizes the linear size of the membrane). There are altogether $N_{bonds} = (3N - 6L)/2$ bonds in the membrane. In our studies we consider *symmetric* hexagonal membranes (*flakes*) so as to minimize possible effects due to the asymmetric of edges or vertices at the membrane periphery.

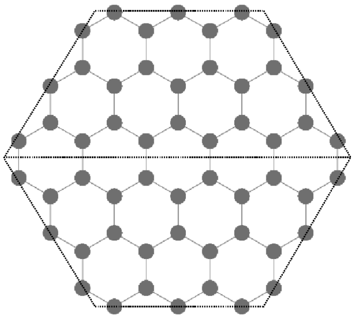

Fig. 1. A model of a membrane with honeycomb structure that contains a total number of $N = 54$ beads and has linear size $L = 3$ (L is the number of beads or hexagonal cells on the edge of the membrane)

C. Potentials. The nearest neighbors in the membrane are connected to each other by "breakable bonds" described by a Morse potential,

$$U_{\mathrm{M}}(r) = \epsilon_M\{1 - \exp[-\alpha(r - r_{\min})]\}^2 \tag{1}$$

where r is a distance between the monomers, $\alpha = 1$ is a constant that determines bond elasticity, $r_{\min} = 1$ is the equilibrium bond length. The dissociation energy of a given bond is $\epsilon_M = 1$, measured in units of k_BT, where k_B denotes the Boltzmann constant and T is the temperature. The minimum of this potential occurs at $r = r_{\min}$, $U_{\mathrm{Morse}}(r_{\min}) = 0$. The maximal restoring force of the Morse potential, $f_{\max} = -dU_{\mathrm{M}}/dr = \alpha\epsilon_M/2$, is reached at the inflection point, $r_{\mathrm{inflex}} = r_{\min} + \alpha^{-1}\ln(2) \approx 2.69$. This force $f_{\max}$ determines the maximal tensile strength of the membrane. Since $U_{\mathrm{M}}(0) \approx 2.95$, the Morse potential, Eq. (1), is only weakly repulsive and beads could partially penetrate one another at $r < r_{\min}$. Therefore, in order to allow properly for the excluded volume interactions between bonded monomers, we take the bond potential as a sum of the Morse potential, $U_{\mathrm{M}}(r)$, and the so called Weeks-Chandler-Anderson (WCA) potential, $U_{\mathrm{WCA}}(r)$, (i.e., the shifted and truncated repulsive branch of the Lennard-Jones potential),

$$U_{\mathrm{WCA}}(r) = \begin{cases} 4\epsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right] + \epsilon, & \text{for } r \leq 2^{1/6}\sigma \\ 0, & \text{for } r > 2^{1/6}\sigma \end{cases} \tag{2}$$

with parameter $\epsilon = 1$ and monomer diameter $\sigma = 2^{-1/6} \approx 0.89$ so that the minimum of the WCA potential coincides with the minimum of the Morse potential. Thus, the length scale is set by the parameter $r_{\min} = 2^{1/6}\sigma = 1$. The nonbonded interactions between monomers are also taken into account by means of the WCA potential, Eq. (2). The nonbonded interactions in our model correspond to good solvent conditions whereas the bonded interactions make the bonds in our model breakable so they undergo scission at sufficiently high temperature T.

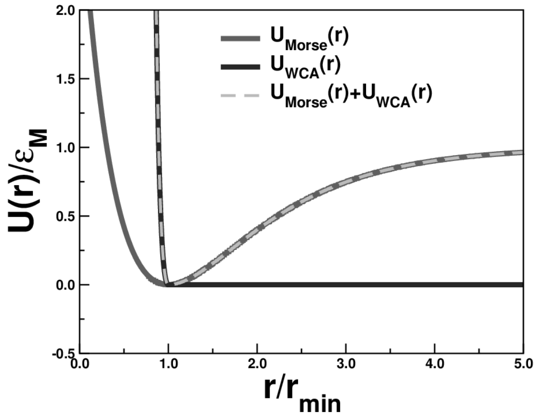

Fig. 2. The interactions between neighboring atoms are described by the following potential model: the nonbonded interactions are described by the Weeks-Chandler-Andersen (WCA) potential (i.e., the shifted and truncated repulsive branch of the Lennard-Jones potential) and the bonded interactions are taken into account as a sum of the Morse potential, Eq. (1), and the WCA potential, Eq. (2)

D. Molecular dynamics simulation procedure. In our MD simulation we use a Langevin dynamics, which describes the Brownian motion of a set of interacting particles whereby the action of the solvent is split into slowly evolving viscous (frictional) force and a rapidly fluctuating stochastic (random) force. The Langevin equation of motion is the following:

$$m\dot{\vec{v_i}}(t) = \vec{F}_i(t) - m\gamma\vec{v_i}(t) + \vec{R}_i(t) \tag{3}$$

where m denotes the mass of the particles which is set to $m = 1$, $\overrightarrow{v}_i$ is the velocity of particle i, $\overrightarrow{F}_i = (\overrightarrow{F}_M + \overrightarrow{F}_{WCA})_i$ is the conservative force which is a sum of all forces exerted on particle i by other particles in the system, γ is the friction coefficient and $\overrightarrow{R}_i$ is the three-dimensional vector of random force acting on particle i. The random force $\overrightarrow{R}_i$, which represents the incessant collision of the monomers with the solvent molecules, satisfies the fluctuation-dissipation theorem $\langle R_{i\alpha}(t) R_{j\beta}(t') \rangle = 2\gamma k_B T \delta_{ij} \delta_{\alpha\beta} \delta(t - t')$ where the symbol $\langle ... \rangle$ denotes an equilibrium average and the Greek-letter subscripts refer to the x, y or z components. The friction coefficient γ of the Langevin thermostat is set to $\gamma = 0.25$. The integration step is 0.002 time units (t.u.) and the time is measured in units of $r_{\min}\sqrt{m/\epsilon_M}$. We emphasize at this point that in our coarse-grained modeling no explicit solvent particles are included. In this work the velocity-Verlet algorithm is used to integrate the equations of motion.

E. Results. Our MD simulations are carried out in the following order. First, we prepare an equilibrated membrane conformation starting with a fully flat configuration (shown schematically in Fig. 1), where each bead in the network is separated by a distance $r_{\min} = 1$ equal to the equilibrium separation of the bond potential $(U_{\mathrm{M}} + U_{\mathrm{WCA}})$ [see Eqs. (1) and (2)]. Then we start the simulation with this prepared conformation and let the membrane equilibrate in the heat bath at a temperature low enough that the membrane stays intact (this equilibration is done in order to prepare different starting conformations for each simulation). Once the equilibration is finished, the temperature is raised to the working one and we let the membrane equilibrate at this temperature for a while (roughly, ~ 20 t.u.). Then the time is set to zero and we continue the simulation of this well-equilibrated membrane conformation (see Fig. 3) to examine the thermal (temperature-induced) scission of the bonds.

In the course of simulation we calculate properties such as the probability distribution of breaking bonds regarding their position in the membrane, the mean first breakage time of a bond (i.e., the elapsed time until the first bond breakage occurs) depending on membrane size and temperature, the probability distribution of the first breakage time, the mean extension of the bonds in the membrane, as well as other quantities of interest. A detailed description of these measurements can be found in our recent paper [62] where we demonstrated that at lower temperature $T = 0.10$ the degradation process starts from the rim of the membrane sheet and then proceeds inwards. In contrast, at higher temperature $T = 0.15$ bonds break at random everywhere in the network sheet. The mean first breakage time τ is found to decrease with the total number of network nodes N by a power law $\tau \propto N^{-0.5}$ and reveals an Arrhenian dependence on temperature,

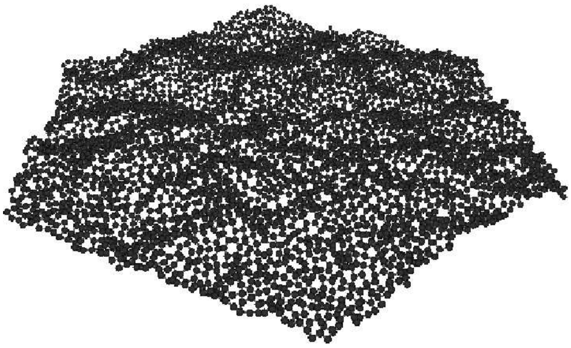

Fig. 3. A snapshot of a typical conformation of an intact membrane with $L = 30$ containing $N = 5400$ monomers after equilibration. Typical ripples are seen to cross the surface

$\tau \propto \exp(\Delta E_b/k_B T)$, with a dissociation energy $\Delta E_b \approx 1$.

Also we examine the course of the degradation kinetics of a honeycomb membrane in this way: at periodic time intervals in separate simulation runs we analyze the distribution of fragment (cluster) sizes n of the initial membrane and establish the time-dependent probability distribution function of fragment sizes, $P(n,t)$, as time elapses after the onset of the thermal degradation process. This also yields the time evolution of the mean fragment size, $N(t) = \int n(t)P(n,t)dn$. Thus, for a given time moment t we average data over more than 10^3 independent runs, each starting from a different initial conformation of the honeycomb membrane. We perform monitoring of the fragmentation process and statistical averaging of fragment sizes using appropriately developed for the system fast cluster counting algorithm (given in the Appendix).

In the next, we present some results of our simulation study concerning the temporal evolution of the fragmentation process. After the onset of the thermal decomposition process the membrane flake disintegrates with time into smaller fragments (clusters) of size n. In Fig. 4 we show the time variation of the ensuing probability distribution of fragment sizes $P(n,t)$. The initial size of the membrane is $N = 294$ monomers and the temperature is $T = 0.12$. The system is seen to start with a single sharp peak at time $t = 0$ when the membrane is still intact. As time goes by, $P(n,t)$ becomes bimodal, the maximum of the distribution is seen to shift to smaller values of cluster size whereas an accumulation of fragments of size 1 or 2 is observed to contribute to a second peak at $n \approx 1$. Eventually, as $t \to \infty$, the probability distribution $P(n,t)$ settles to a shape with a single sharp

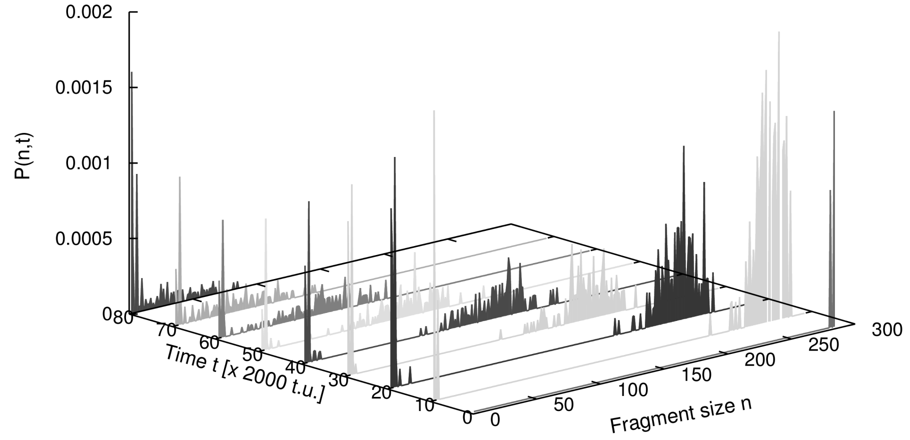

Fig. 4. Probability distribution of fragment (cluster) sizes $P(n,t)$ at different times t (in MD time units) after beginning of the thermal degradation process for a membrane with $N = 294$ monomers. The parameters of the heat bath are $T = 0.12$ and $\gamma = 0.25$

peak (a δ-function) at $n \approx 1$.

In Fig. 5 we give the course of temporal evolution of the mean fragment size $N(t)$, observed in our computer experiment for membranes with different initial size N_0 (i.e., N_0 is the initial number of monomers when the membrane is still intact). It can be seen that after the onset of the thermal degradation process the membrane network disintegrates with time into smaller fragments whose mean size (or average molecular weight) $N(t)$ decreases steadily with time. One can readily see that the quantity $1 - N^{-1}(t)$ does not immediately follow a straight line of decay when plotted in semi-logarithmic coordinates, rather, such a linear decay is observed after an initial period of slower decline. This effect is due to averaging over many realizations of the fragmentation process. In each run the degradation of bonds starts earlier or later at a particular time τ (the mean first breakage time). As a result a clear-cut exponential course of $1 - N^{-1}(t)$ is only observed in the late stages of fragmentation. Such exponential behavior is found independently of the initial membrane size.

In addition, one could expect that the fragmentation process is not governed by a single rate constant in a presumably 1st-order chemical reaction even though the bonds that undergo rupture are chemically identical. Therefore, from the temporal mean cluster size behavior, presented in Fig. 5, one may conclude that even in the case of a homogeneous membrane the thermal degradation process

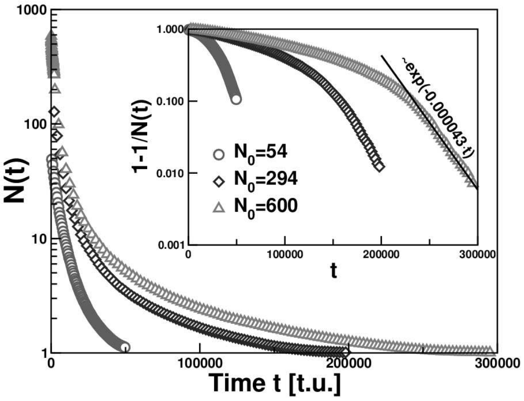

Fig. 5. Semi-logarithmic plot of time variation of mean fragment size $N(t)$ for membranes with different initial size N_0 (as indicated in the legend). In the inset the same is shown for the quantity $1 - 1/N(t)$. Symbols represent simulation results whereas a straight line stands for the fitting function $1 - 1/N(t) \propto \exp(-kt)$, with a kinetic constant $k = 4.3 \times 10^{-5}$. The parameters of the heat bath are $T = 0.12$ and $\gamma = 0.25$

is more adequately described by several reaction constants which govern the dissociation of different groups of bonds. In the recent paper [62] we suggest a simple model of reaction kinetics which takes into account this conjecture. We developed a theoretical scheme based on a set of 1st-order kinetic differential equations, describing the variation of the number of network nodes, connected by a particular number of bonds to neighboring nodes, as time elapses. We demonstrated that the analytical solution of such a system provides a faithful description of fragmentation kinetics. More details about the simulation and more results concerning the thermal decomposition of a honeycomb membrane can be found in our recent paper [62].

5. Summary. A careful construction of efficient computer algorithms is of prime importance for treating large samples. In this paper we describe a simplified implementation of the Hoshen-Kopelman cluster counting algorithm adapted for honeycomb networks. The description of the algorithm is also translated into the C programming language in the Appendix. In the implementation of the algorithm we assume that all nodes in the network are occupied and links between nodes can be intact or broken. The algorithm counts the total number of clusters in the network and performs cluster statistics. To determine the probability distribution of cluster sizes, we calculate the number of clusters of each size. The cluster-counting technique is applied in conjunction with Langevin Molecular dynamics simulation of a honeycomb membrane to investigate the fragmentation kinetics and the distribution of fragment sizes as time elapses during the thermal destruction process. Using an *ad hoc* cluster counting program in the course of the MD-simulation we sample the time variation of the probability distribution of fragment sizes, $P(n,t)$, which gives the time evolution of the mean cluster size $N(t) = \displaystyle\int n(t)P(n,t)dn$. The distribution of fragments sizes evolves with elapsed time from initially a δ-function through a bimodal one into a single-peaked again at late times. The proposed implementation of the Hoshen-Kopelman cluster counting algorithm is useful for application to a large system with a honeycomb (hexagonal) structure, but it can be easily changed to be applied for networks with an arbitrary connectivity between the nodes (triangular, square, and so on structure).

Acknowledgments. The author is grateful for the financial help of Project BG051PO001-3.3.06-0038 funded by OP Human Resources Development 2007-2013 of EU Structural Funds. The author also acknowledges the use of the high performance computing facility MADARA at the Bulgarian Academy of Sciences. In addition the author would like to thank Prof. Andrey Milchev for his useful suggestions and discussions.

REFERENCES

[1] Broadbent S. R., J. M. Hammersley. Percolation processes I. Crystals and mazes. *Proc. Camb. Phil. Soc.* **53** (1957), 629–641.

[2] Hoshen J., R. Kopelman. Percolation and cluster distribution I. Cluster multiple labeling technique and critical concentration algorithm. *Phys. Rev. B*, **14** (1976), No 8, 3438–3445.

[3] Hammersley J. M. Origins of percolation theory. In: Percolation structures and processes, (Eds G. Deutscher, R. Zallen, J. Adler), Adam Hilger, Bristol, 1983, 47–57.

[4] Stauffer D., A. Aharony. Introduction to percolation theory. Taylor and Francis, London, 1992.

[5] Margolina A., H. Nakanishi, D. Stauffer, H. E. Stanley. Monte Carlo and series study of corrections to scaling in two-dimensional percolation. *J. Phys. A*, **17** (1984), 1683–1701.

[6] Rapaport D. C. Cluster number scaling in two-dimensional percolation. *J. Phys. A*, **19** (1986), 291–304.

[7] Rapaport D. C. Cluster size distribution at criticality. *J. Stat. Phys.*, **66** (1986), 679–682.

[8] Aho A. V., J. E. Hopcroft, J. D. Ullman. The Design and Analysis of Computer Algorithms. Addison-Wesley, Reading, MA, 1974.

[9] Babalievski F. Cluster counting: the Hoshen-Kopelman algorithm versus spanning tree approaches. *Int. J. Mod. Phys. C*, **9** (1998), No 1, 43–60.

[10] Eddi F., J. Mariani, G. Waysand. Transient synaptic redundancy in the developing cerebellum and isostatic random stacking of hard spheres. *Biol. Cybern.*, **74** (1996), No 2, 139–146.

[11] Hamimov S., M. A. Michalev, A. Savchenko, O. I. Yordanov. Classification of radar signatures by autoregressive model-fitting and cluster-analysis. *IEEE Trans. Geosci. Remote Sens.*, **27** (1989), No 5, 606–610.

[12] Kinney J. H., D. L. Haupt, M. C. Nichols, T. M. Breunig, G. W. Marshall, S. J. Marshall. The X-Ray tomographic microscope: 3-dimensional perspectives of evolving microstructures. *Nucl. Instrum. Methods A*, **347** (1994), No 1-3, 480–486.

[13] Moreira R. G., M. A. Barrufetb. Spatial distribution of oil after deep-fat frying of tortilla chips from a stochastic model. *J. Food Eng.*, **27** (1996), No 3, 279–290.

[14] Zhang L., N. A. Seaton. Simulation of catalyst fouling at the particle and reactor levels. *Chem. Eng. Sci.*, **51** (1996), No 12, 3257–3272.

[15] Frisch H. L., J. M. Hammersley. Percolation Processes and Related Topics. *J. Soc. Ind. Appl. Math. B*, **11** (1963), 894–918.

[16] Shante V. K. S., S. Kirkpatrick. An introduction to percolation theory. *Adv. Phys.*, **20** (1971), 325–357.

[17] Essam J. W. Percolation and cluster size. In: Phase Transition and Critical Phenomena (Eds C. Domb, M. S. Green), Academic Press, New York, 1972, Vol **2**, 197–270.

[18] Halperin B. I., D. J. Bergman. Heterogeneity and disorder: Contributions of Rolf Landauer. *Physica B*, **405** (2010), 2908–2914.

[19] Shearing P. R., D. J. L. Brett, N. P. Brandon. Towards the Intelligent Engineering of SOFC Electrodes: A Review of advanced microstructural characterisation techniques. *International Materials Reviews*, **55** (2010), 347–363.

[20] Silva J., R. Simoes, S. Lanceros-Mendez, R. Vaia. Applying complex network theory to the understanding of high-aspect-ratio carbon-filled composites. *Europhys. Lett.*, **93** (2011), 37005 p1–37005 p6.

[21] Suzuki S. U., A. Sasaki. How does the resistance threshold in spatially explicit epidemic dynamics depend on the basic reproductive ratio and spatial correlation of crop genotypes? *J. Theor. Biol.*, **276** (2011), 117–125.

[22] Lindquist J., Junling Ma, P. van den Driessche, F. Willeboordse. Effective degree network disease models. *J. Math. Biol.*, **62** (2011), 143–164.

[23] Naumova E. N., J. Gorski, Yu. N. Naumov. Simulation studies for a multistage dynamic process of immune memory response to influenza: experiment in silico. *Annales Zoologici Fennici*, **45** (2008), 369–384.

[24] Floyd W., L. Kay, M. Shapiro. Some elementary properties of SIR networks or, can i get sick because you got vaccinated? *Bull. Math. Biol.*, **70** (2008), 713–727.

[25] Camelo-Neto G., S. Coutinho. Forest-fire model with resistant trees. *J. Stat. Mech.*, 2011. doi: 10.1088/1742-5468/2011/06/P06023

[26] Guisoni N., E. S. Loscar, E. V. Albano. Phase diagram and critical behavior of a forest-fire model in a gradient of immunity. *Phys. Rev. E*, **83** (2011), 011125.
http://dx.doi.org/10.1103/PhysRevE.83.011125

[27] Simeoni A., P. Salinesi, F. Morandini. Physical Modelling of Forest Fire Spreading through Heterogeneous Fuel Beds. *Int. J. Wildland Fire*, **20** (2011), 625–632.

[28] Amiaz Y., S. Sorek, Y. Enzel, O. Dahan. Solute transport in the vadose zone and groundwater during flash floods. *Water Resources Research*, **47** (2011). doi: 10.1029/2011WR010747

[29] Bolandtaba S. F., A. Skauge. Network Modeling of EOR Processes: A Combined Invasion Percolation and Dynamic Model for Mobilization of Trapped Oil. *Transport in Porous Media*, **89** (2011), 357–382.

[30] Mourzenko V. V., J.-F. Thovert, P. M. Adler. Permeability of isotropic and anisotropic fracture networks, from the percolation threshold to very large densities. *Phys. Rev. E*, **84** (2011). http://dx.doi.org/10.1103/PhysRevE.84.036307

[31] Torelli L., A. E. Scheidegger. Random maze models of flow through porous media. *Pure Appl. Geophys.*, **89** (1971), 32–44.

[32] De Gennes P. G., E. Guyon. Lois générales pour l'injection d'un fluide dans un milieu poreux aléatoire. *Journal de Mécanique*, **17** (1978), 403–432.

[33] Ganjeh-Ghazvini M., M. Masihi, M. Ghaedi. Random walk-percolation-based modeling of two-phase flow in porous media: Breakthrough time and net to gross ratio estimation. *Physica A*, **406** (2014), 214–221.

[34] Stauffer D. Gelation in concentrated critically branched polymer solutions. Percolation scaling theory of intramolecular bond cycles. *J. Chem. Soc. Faraday Trans. II*, **72** (1976), 1354–1364.

[35] Elliot R. J., B. R. Heap, D. J. Morgan, G. S. Rushbrooke. Equivalence of the Critical Concentrations in the Ising and Heisenberg Models of Ferromagnetism. *Phys. Rev. Lett.*, **5** (1960), 366–367.

[36] Pike G. E., W. J. Camp, C. H. Seager, G. L. McVay. Percolative aspects of diffusion in binary alloys. *Phys. Rev. B*, **10** (1974), 4909–4917.

[37] Kikuchi R., H. Sato. Correlation Factor in Substitutional Diffusion in Binary Alloys. *J. Chem. Phys.*, **53** (1970), 2702–2713.

[38] Hoshen J., R. Kopelman. Exciton percolation I. Migration dynamics. *J. Chem. Phys.*, **65** (1976), 2817–2823.

[39] Kopelman R., E. M. Monberg, F. W. Ochs, P. N. Prasad. Exciton Percolation: Isotopic-Mixed $^1B_{2u}$ Naphthalene. *Phys. Rev. Lett.*, **34** (1975), 1506–1509.

[40] Kirkpatrick S. Classical Transport in Disordered Media: Scaling and Effective-Medium Theories. *Phys. Rev. Lett.*, **27** (1971), 1722–1725.

[41] Seager C. H., G. E. Pike. Percolation and conductivity: A computer study. II, *Phys. Rev. B*, **10** (1974), 1435–1446.

[42] Cohen M. H., I. Webman, J. Jortner. Optical and microwave properties of metal-ammonia solutions. *J. Chem. Phys.*, **64** (1976), 2013–2019.

[43] Pollak M., I. Reiss. A percolation treatment of high-field hopping transport. *J. Phys. C*, **9** (1976), 2339–2352.

[44] Campos P. R. A., L. F. C. Pessoa, F. G. Brady Moreira. Cluster-size statistics of site-bond-correlated percolation models. *Phys. Rev. B*, **56** (1997), No 1, 40–42.

[45] Hoshen J. Percolation analog for a two component liquid vapor system. *Chem. Phys. Lett.*, **75** (1980), 347–349.

[46] Hoshen J., P. Klymko, R. Kopelman. Percolation and cluster distribution III. Algorithm for the site-bond problem. *J. Stat. Phys.*, **21** (1979), No 5, 583–600.

[47] González M. I., P. Centres, W. Lebrecht, A. J. Ramirez-Pastor, F. Nieto. Site-bond percolation on triangular lattices: Monte Carlo simulation and analytical approach. *Physica A*, **392** (2013), 6330–6340.

[48] Hoshen J., M. W. Berry, K.S. Minser. Percolation and cluster structure parameters: the enhanced Hoshen-Kopelman algorithm. *Phys. Rev. B*, **56** (1997), No 2, 1455–1460.

[49] Hoshen J. On the application of the enhanced Hoshen-Kopelman algorithm for image analysis. *Pattern Recogn. Lett.*, **12** (1998), 575–584.

[50] Tsang I. J., I. R. Tsang, D. Van Dyck. Cluster diversity and entropy on the percolation model: the lattice animal identification algorithm. *Phys. Rev. B*, **62** (2000), No 5, 6004–6014.

[51] Burkitt A. N., D. W. Heermann. Parallelization of a cluster algorithm, *Comput. Phys. Commun.* **54** (1989), No 2-3, 201–209.

[52] Constantin J. M., M. W. Berry, B. T. Vander Zanden. Parallelization of the Hoshen-Kopelman algorithm using a finite state machine. *Int. J. Supercomput. Appl. High Perform. Comput.*, **11** (1997), No 1, 34–48.

[53] Flanigan M., P. Tamayo. Parallel cluster labeling for large-scale Monte-Carlo simulations. *Physica A*, **215** (1995), No 4, 461–480.

[54] Teuler J. M., J. C. Gimel. A direct parallel implementation of the Hoshen-Kopelman algorithm for distributed memory architectures. *Comput. Phys. Commun.*, **130** (2000), No 1–2, 118–129.

[55] Frijters S., T. Krüger, J. Harting. Parallelised Hoshen-Kopelman algorithm for lattice-Boltzmann simulations. *Comput. Phys. Commun.*, **189** (2015), 92–98.

[56] Gawlinski E. T., H. E. Stanley. Continuum percolation in two dimensions: Monte Carlo tests of scaling and Universality for non-interacting discs. *J. Phys. A* **14** (1981), L291–L299.

[57] Lee S. B., S. Toruato. Monte Carlo study of correlated continuum percolation: Universality and percolation thresholds. *Phys. Rev. A*, **41** (1990), No 10, 5338–5344.

[58] Lorenz B., I. Orgzall, H. O. Heuer. Universality and cluster structures in continuum models of percolation with two different radius distributions. *J. Phys. A*, **26** (1993), 4711–4722.

[59] Strogatz S. H. Exploring complex networks. *Nature*, **410** (2001), 268–276.

[60] Koopman E. A., C. P. Lowe. An algorithm for detecting percolating structures in periodic systems - Application to polymer networks. *Journal of Computational Physics*, **274** (2014), 758–769.

[61] Al-Futaisi A., T. W. Patzek. Extension of Hoshen-Kopelman algorithm to non-lattice environments. *Physica A*, **321** (2003), 665–678.

[62] Paturej J., H. Popova, A. Milchev, T. A. Vilgis. Thermal decomposition of a honeycomb-network sheet: A molecular dynamics simulation study. *J. Chem. Phys.*, **137** (2012).
http://dx.doi.org/10.1063/1.4739536

[63] Paturej J., H. Popova, A. Milchev, T. A. Vilgis. Force-induced breakdown of flexible polymerized membrane. *Phys. Rev. E*, **85** (2012).
http://dx.doi.org/10.1103/PhysRevE.85.021805

[64] Allen N. S., M. Edge. Fundamentals of Polymer Degradation and Stabilization. Elsevier Applied Science, New York, 1966.

[65] Madras G., J. M. Smith, B. J. McCoy. Degradation of poly (methyl methacrylate) in solution. *Ind. Eng. Chem. Res.,* **35** (1996), 1795–1800.

[66] Nyden M. R., G. P. Forney, G. E. Brown. Molecular modeling of polymer flammability: application to the design of flame-resistant polyethylene. *Macromolecules*, **25** (1992), 1658–1666.

[67] Sijbesma R. P., F. H. Beijer, L. Brunsveld, B. J. B. Folmer, J. H. K. K. Hirschberg, R. F. Lange, J. K. L. Lowe, E. W. Meijer. Reversible Polymers Formed from Self-Complementary Monomers Using Quadruple Hydrogen Bonding. *Science*, **278** (1997), 1601–1604.

[68] Sijbesma R. P., E. W. Meijer. Quadruple hydrogen bonded systems. *Chem. Commun.,* **1** (2003), 5–16.

[69] Neek-Amal M., F. M. Peeters. Linear reduction of stiffness and vibration frequencies in defected circular monolayer graphene. *Phys. Rev. B,* **81** (2010). http://dx.doi.org/10.1103/PhysRevB.81.235437

[70] Sarikaya M., C. Tamerler, A. K. Jen, K. Schulten, F. Banyex. Molecular biomimetics: nanotechnology through biology. *Nature Mater.,* **2** (2003), 577–585.

[71] Han T. H., J. Kim, J. S. Park, C. B. Park, H. Ihee, S. O. Kim. Liquid crystalline peptide nanowires. *Adv. Mater.,* **19** (2007), 3924–3927.

[72] Lindahl T. Instability and decay of the primary structure of DNA. *Nature (London)*, **362** (1993), 709–715.

[73] Sinha R. P., D.-P. Häder. UV-induced DNA damage and repair: a review. *Photochem. Photobiol. Sci.,* **1** (2002), 225–236.

[74] Metanomski W. V., R. E. Bareiss, J. Kahovec, K. L. Loening, L. Shi, V. P. Shibaev. Nomenclature of Regular Double-Strand (Ladder and Spiro) Organic Polymers. *Pure Appl. Chem.,* **65** (1993), 1561–1580.

[75] Jellinek H. H. G. On the degradation of long chain molecules. Part I. *Trans. Faraday Soc.,* **40** (1944), 266–273.

[76] Ballauff M., B. A. Wolf. Degradation of chain molecules. 1. Exact solution of the kinetic equations. *Macromolecules.* **14** (1981), 654–658.

[77] Blaisten-Barojas E., M. R. Nyden. Molecular dynamics study of the depolymerization reaction in simple polymers. *Chem. Phys. Lett.,* **171** (1990), 499–505.

[78] Nyden M. R., D. W. Noid. Molecular dynamics of initial events in the thermal degradation of polymers. *J. Phys. Chem.,* **95** (1991), 940–945.

[79] Hathorn B. C., B. G. Sumpter, D. W. Noid. On the distribution of fragment sizes in the fragmentation of polymer chains. *Macromol. Theory Simul.,* **10** (2001), 587–591.

[80] Doruker P., Y. Wang, W. L. Mattice. Simulation of the random scission of C-C bonds in the initial stage of the thermal degradation of polyethylene. *Comp. Theor. Polym. Sci.,* **11** (2001), 155–166.

[81] Paturej J., A. Milchev, V. G. Rostiashvili, T. A. Vilgis. Thermal degradation of unstrained single polymer chain: non-linear effects at work. *J. Chem. Phys.,* **134** (2011), 224901.

[82] Hansen J.C., R. Skalak, S. Chien, A. Hoger. An elastic network model based on the structure of the red blood cell membrane skeleton. *Biophys. J.,* **70** (1996), 146–166.

[83] Beale P.D., D.J. Srolovitz. Elastic fracture in random materials. *Phys. Rev. B,* **37** (1988), 5500–5507.

[84] Saxton M. J. The membrane skeleton of erythrocytes. A percolation model. *Biophys. J.,* **57** (1990), 1167-1177.

[85] Boal D. H., U. Seifert, A. Zilker. Dual network model for red blood cell membranes. *Phys. Rev. Lett.,* **69** (1992), 3405-3408.

[86] Monette L., M. P. Anderson. Elastic and fracture properties of the two-dimensional triangular and square lattices. *Modelling Simul. Mater. Sci. Eng.,* **2** (1994), 53–66.

[87] Dao M., J. Li, S. Suresh. Molecularly based analysis of deformation of spectrin network and human erythrocyte. *Mater. Sci. Eng. C,* **26** (2006), 1232–1244.

[88] Barnard A. S., I. K. Snook. Thermal stability of graphene edge structure and graphene nanoflakes. *J. Chem. Phys.,* **128** (2008), 094707-1–094707-7.

[89] Argyropoulos D. S., H. I. Bolker. The gel degradation theory, Part I. An experimental verification with a model trifunctional network. *Macromolecules,* **20** (1987), 2915-2922.

[90] Argyropoulos D. S., H. I. Bolker. The gel degradation theory, Part II. An experimental verification with model networks formed by the random crosslinking of monodisperse primary chains. *Macromol. Chem.,* **189** (1988), 607–618.

[91] Barral L., F. J. Diez, S. Garcia-Garabal, J. Lopez, B. Montero, R. Montes, C. Ramirez, M. Rico. Thermodegradation kinetics of a hybrid inorganic-organic epoxy system. *Europ. Polym. J.,* **41** (2005), 1662–1666.

[92] Zhang Z., G. Liang, P. Ren, J. Wang. Thermodegradation kinetics of epoxy/DDS/POSS system. *J. Polym. Composites,* **28** (2007), 755–761.

[93] Chenoweth K., S. Cheung, A. C. T. van Duin, W. A. Goddard III, E. M. Kober. Simulations on the thermal decomposition of a poly(dimethylsiloxane) polymer using the ReaxFF reactive force field. *J. Am. Chem. Soc.,* **127** (2005), 7192–7202.

APPENDIX: C function for cluster counting

Here, we provide the complete version of the function **CLUSTER_COUNT()** written in the C programming language that performs our implementation of the Hoshen-Kopelman cluster counting algorithm adapted for honeycomb networks in non-lattice environments. Using the data structures proposed in this paper as an input, the function performs cluster statistics. First, the function **CLUSTER_COUNT()** finds how many clusters there are in the network and determines which nodes belong to each cluster. Then the function counts the number of clusters of each size and determines the probability distribution of cluster sizes. In addition, there are defined the auxiliary functions **min2(x,y)** and **min3(x,y,z)**, which return the minimum among two or three integer numbers, respectively.

```
/****************************  AUXILIARY FUNCTIONS ***************************/
// This function returns the minimum of two integer numbers
#define min2(x,y) ((x<y)?x:y)
```

```
// This function returns the minimum of three integer numbers
#define min3(x,y,z) ((x<y&&x<z)?x:(y<z)?y:z)
/******************************************************************************/

void CLUSTER_COUNT(long N)
{
 /*=============================================================================
 Adaptation of Hoshen-Kopelman cluster counting algorithm for honeycomb networks
 =============================================================================*/
    /* Local variables */
    long i, nn1, nn2, nn3, link1, link2, link3, L_nn1, L_nn2, L_nn3;
    long cluster_counter, cn, i_cn, NodeLPmin;
    long NumberOfClusters, max_cluster_number, cluster_number, num_mono;
    long Num_cl[4];
 /*=============================================================================
    Input arguments:
      N                  - Number of nodes (monomers) in the network
      Bond_Neighbour[][] - Neighboring nodes connected to each node
      bond_rupture[i][j] - State of the bond (0-intact/1-broken) between
                           monomers i and j
    Output arguments:
      NumberOfClusters   - Number of occupied clusters
      NodeL[]            - Array to store cluster labels of nodes
      NodeLP[]           - Array that holds information about the cluster labels
 =============================================================================*/

 /****************  COUNT CLUSTERS IN A NETWORK CONFIGURATION  ****************/

 // STEP 1: READ THE DATA AND INITIALIZE THE OUTPUT
    NumberOfClusters  = 0;

 // STEP 2: INITIALIZE THE HOSHEN-KOPELMAN ALGORITHM VARIABLES - array NodeL
    for (i = 0; i < N; i++) NodeL[i] = 0;

 // STEP 3: CREATE EMPTY ARRAY NodeLP AND START CLUSTER COUNTER
    for (i = 0; i <= N; i++) NodeLP[i]=0;
    cluster_counter=0; // Cluster counter

 // STEP 4: SCAN THE NETWORK NODES
    //i - the number of the current monomer
    //nn1, nn2, nn3 - the numbers of the bonded neighbors of the current monomer

    for (i = 0; i < N; i++) {  //loop over all monomers
      //take the bonded neighbors of the current i-monomer
      nn1 = Bond_Neighbour[i][0];  //take the number of the 1st-bonded neighbor
      nn2 = Bond_Neighbour[i][1];  //take the number of the 2nd-bonded neighbor
      nn3 = Bond_Neighbour[i][2];  //take the number of the 3th-bonded neighbor
```

```
//chek for bond ruptures of i-monomer: link=1-broken bond (bond rupture);
//                                      link=0-intact (unbroken) bond;
link1 = bond_rupture[i][nn1];
link2 = bond_rupture[i][nn2];
if (nn3 != -1) {  //(nn3=-1) when the periphery monomer has only 2 bonded
                  //         neighbors (instead of 3)
   link3 = bond_rupture[i][nn3];
}
else link3 = 1;

// If this node is type (4.1) - all its links are broken!
if (link1==1 && link2==1 && link3==1)
{
  cluster_counter +=1; // Start a new cluster
  NodeL[i]=cluster_counter;
  NodeLP[cluster_counter]=cluster_counter;
} // end if (end case 4.1)

// This node is of type (4.2) - it has at least one intact link!
else
{
  L_nn1 = NodeL[nn1];
  L_nn2 = NodeL[nn2];
  if(nn3 != -1) L_nn3 = NodeL[nn3]; else L_nn3 = 0;

  // Case 4.2 (i): No neighbour is already labeled
  if (L_nn1==0 && L_nn2==0 && L_nn3==0)
  {
    cluster_counter +=1; // Start a new cluster
    NodeL[i]=cluster_counter;
    NodeLP[cluster_counter]=cluster_counter;
  } // end if (end case 4.2 (i))

  // Case 4.2 (ii): There exists a labeled neighbour
  else
  {
    for (cn=0; cn<4; cn++)  Num_cl[cn] = 0;
    cn = 0;
    //if the neighbour is already labeled (so it points to some cluster)
    //   and if it has an unbroken link with current monomer
    if (L_nn1!=0 && link1!=1) Num_cl[++cn] = NodeLP[L_nn1];
    if (L_nn2!=0 && link2!=1) Num_cl[++cn] = NodeLP[L_nn2];
    if (L_nn3!=0 && link3!=1) Num_cl[++cn] = NodeLP[L_nn3];

    if (cn==0) {
      cluster_counter +=1; // Start a new cluster
```

```
           NodeL[i]=cluster_counter;
           NodeLP[cluster_counter]=cluster_counter;
         }
         else { // Put in the minimum labeling
           if (cn==1) NodeLPmin = NodeLP[Num_cl[1]];
           if (cn==2) NodeLPmin = min2(NodeLP[Num_cl[1]], NodeLP[Num_cl[2]]);
           if (cn==3) NodeLPmin = min3(NodeLP[Num_cl[1]], NodeLP[Num_cl[2]],
              NodeLP[Num_cl[3]]);

           NodeL[i] = NodeLPmin;
           for (i_cn=1; i_cn<=cn; i_cn++) NodeLP[Num_cl[i_cn]] = NodeLPmin;
         }
       } // end else (end case 4.2 (ii))
     } // end else (end case 4.2)
   } // end for i

// STEP 5: RENUMBER LABELS IN NodeLP TO RUN SEQUENTIALLY
   max_cluster_number = 0;
   for (i=1; i<=cluster_counter; i++) {
     if(NodeLP[i] > max_cluster_number) {
       max_cluster_number ++;
       NodeLP[i] = max_cluster_number;
     }
   }

// STEP 6: APPLY THE CORRECT LABELS TO THE ARRAY NodeL
   for (i=0; i<N; i++) {
     if (i!=0) NodeL[i] = NodeLP[NodeL[i]];
   }

// RECORD NUMBER OF CLUSTERS
   //NumberOfClusters - total number of clusters in that configuration
   NumberOfClusters = max_cluster_number;

/***********************  PERFORM CLUSTER STATISTICS  ************************/

   //cluster_size[i] - gives the number of monomers in cluster_number "i"
   //NC[i] - gives the total number of clusters of size "i"

   for (i = 0; i <= N; i++) {NC[i] = 0; cluster_size[i] = 0;}

   for (i = 0; i < N; i++) {
     if (NodeL[i] != 0) {
       cluster_number = NodeL[i];
       cluster_size[cluster_number] += 1;
     }
   }
```

```
    for (i = 1; i <= NumberOfClusters; i++) {
      num_mono = cluster_size[i];
      NC[num_mono] += 1.0;
    }

    //create histogram NC = probability distribution of cluster sizes
    for(i=1; i<=N; i++) NC[i] /= NumberOfClusters;

    return ;
}
/*============================================================================*/
```

Hristina Popova
Institute of Physical Chemistry
Bulgarian Academy of Sciences
Acad. G. Bonchev Str, Bl. 11
1113 Sofia, Bulgaria
e-mail: `karleva@ipc.bas.bg`

Received October 15, 2014
Final Accepted May 18, 2015