ELSEVIER

Chemical Physics 275 (2002) 261–269

Chemical Physics

www.elsevier.com/locate/chemphys

Diffusion and trapping in dendrimer structures

Dimitris Katsoulis[a], Panos Argyrakis[a,*], Alexander Pimenov[b], Alexei Vitukhnovsky[b]

[a] *Department of Physics, University of Thessaloniki, 54006 Thessaloniki, Greece*
[b] *Lebedev Physical Institute, Lebedev Physics Research Center, Russian Academy of Sciences, 117924 Moscow, Russia*

Received 7 May 2001; in final form 26 September 2001

Abstract

We investigate diffusion on models of large molecules with dendrimer structure. The models we use are topological Cayley trees. We focus on diffusion properties, such as the excursion distance, the mean square displacement of the diffusing particles, and the area probed, as given by the random walk parameter S_N, the number of the distinct sites visited. Finally we look at the trapping kinetics for randomly distributed stationary traps of low concentrations. We use different coordination numbers, z, and different generation orders g of a dendrimer structure. We show that for calculations on dendrimers where we simulate the entire structure the finite size effects dominate making the results worthless. We, therefore, implemented two algorithms that do not retain the underlying structure, and thus effectively the random walk is performed on an equivalent infinite structure. We find a universal linear behavior for all the properties examined, thus differing from the regular lattices. The average displacement R grows linearly with time, implying that R^2 grows as t^2 instead of t. The S_N property also grows linearly with time, which resembles the random walk case in three-dimensional lattices, but with a different prefactor which strongly depends on z. Trapping is dependent on S_N, as expected, and follows exactly its behavior. Due to the nature of these structures the random walk is indeed a type of a "biased" walk. We devise a model in which we take out this "bias" factor, and in this model we show that the system becomes diffusive in nature, i.e. the mean square displacement is linear with time. These results could be utilized if one is to design a dendrimer system with specifically desired properties.

1. Introduction

Diffusion properties on structures with Cayley tree topology have recently attracted a considerable interest mainly because of the new synthesis of dendrimer molecules [1–8], thus becoming ideal model structures having exactly the required geometry of the actual molecular systems. The molecules synthesized are characterized by a core, and several generations of emanating branches, as shown in Fig. 1. Varying the branching ratio z (or number of nearest neighbors, also known as coordination number), and the generation number (order) g can produce a large variety of such structures. It has also been proposed that controlling these two parameters is the key to

* Corresponding author.
E-mail address: panos@physics.auth.gr (P. Argyrakis).

0301-0104/02/$ - see front matter
PII: S0301-0104(01)00524-9

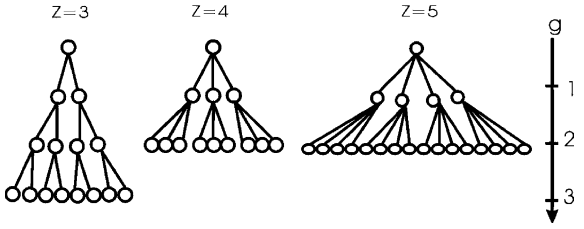

Fig. 1. Schematic of three dendrimers, with $z = 3$, 4, and 5. The generation order, g, is as shown, $g = 3$ (for $z = 3$), and $g = 2$ (for $z = 4$, and 5)

controlling energy transport in such structures. It is thus of interest to see what role do these play, search for ideal combinations of z and g, etc. There is now evidence that structures that have very small differences in these parameters that they can exhibit very different behavior in their transport properties. For example, the ''compact'' and ''extended'' series in repeated phenylacetylene units exhibit enhanced rates, the first one towards the core, but the second towards the opposite direction, the molecular periphery, even though both systems are composed of identical single phenylacetylene chains, but with slightly different molecular layouts [7,8].

Furthermore, from the geometrical point of view, Cayley trees have been well known for a long time, and are unique models of structures in which the inherent notion of dimensionality does not exist. By the usual way of thinking, if one considers the abstract Cayley model, their dimensionality is infinity. But, of course, actual dendrimer molecules would be embedded in normal three-dimensional space. Thus, at least from the theoretical point of view, it is interesting to investigate well known diffusion properties in order to compare them to the equivalent processes in the classical dimensionalities of lattice systems. In a sense dendrimers belong to the so called class of complex systems which have shown very rich, albeit different and/or peculiar behavior in the past decades.

By the nature of the dendrimer structures, the dynamical properties that are investigated on them are highly asymmetric. Transfer towards the periphery has a higher overall probability, when compared to transfer towards the core of the structure, for any number of branches. In a corresponding random walk model, which is frequently used for such calculations, this type of diffusion is analogous to biased or directional diffusion, in which the probability for moving to the right is not the same as the probability to move to the left (for a one-dimensional case). It is thus expected that this will lead to a ballistic-type motion, rather than a purely stochastic one, with the average excursion being proportional to N rather than $N^{1/2}$ (N is the number of steps, or time), as is the case for any type of lattice system, where this bias does not exist.

In the present paper we deal in detail with random walks on dendrimer structures by investigating the average displacement of the walk, $\langle R \rangle$. Since motion is ballistic we expect that $\langle R^2 \rangle$ would be proportional to N^2 since $\langle R \rangle$ is expected to be proportional to N. We then look at S_N, the number of distinct sites visited at least once, as a function of time. It is well known that this property has a different behavior in lattices of different dimensionalities, i.e. for one-dimensional systems $S_N \propto N^{1/2}$ for two-dimensional ones $S_N \propto N/\log N$, and for three-dimensional ones $S_N \propto N$. Due to the ballistic character of the walk one would expect a behavior of $S_N \propto N$, i.e. similar to three-dimensional systems. There is a theoretical asymptotic prediction for S_N on the dendrimers which gives exactly this result [9]:

$$S_N = \frac{z-2}{z-1} N, \quad N \to \infty. \tag{1}$$

Finally, a related property in the same class of random walk properties is that of trapping. The problem of trapping has been known for a long time [10], but a general finite-time solution is still lacking, with the exception of one-dimensional lattices. In this problem traps of small concentration are randomly distributed on the dendrimer nodes. One monitors the time that it takes for a particle performing a random walk to be trapped irreversibly in one of the trap sites. From the distribution of the trapping times we can derive the survival probability, Ψ_N. An asymptotic solution for Ψ_N due to Donsker and Varadhan [11] is valid for any dimensionality d, and actually contains d as a parameter in exponential form. This means

that the problem is extremely sensitive to the dimensionality, and as such it would be interesting to study its behavior in dendrimers, where there is no apparent d. Obviously, the Donsker–Varadhan form cannot be used here. Also, this solution is valid for time $t \to \infty$ so that nobody has ever numerically calculated the Donsker–Varadhan behavior. As it has been shown by Rosenstock [12], at least for early times, trapping is dominated by S_N, irrespective of the underlying system. This is an approximation only, but for small times it is a fairly good one. In this approximation the survival probability is given by:

$$\Psi_N = e^{-\lambda S_N}, \tag{2}$$

where $\lambda = -\ln(1-C)$, and C is the trap concentration. Thus, at least within this approximation, we can avoid the difficulty of solving analytically the Donsker–Varadhan equation and use the S_N directly from Eq. (1), in order to derive the survival probabilities.

A new characteristic that we find very important in computer simulations in the dendrimer structures is that due to their geometry finite size effects are always prominent. The size of a dendrimer grows as a geometrical progression, according to the formula:

$$M = \sum_{i=0}^{g} (z-1)^i = \frac{(z-1)^g - 1}{z-2}, \tag{3}$$

where M is the total number of sites (nodes), g is the generation order, and z is the coordination number. We see that M grows extremely fast. For example, for $z = 3$, and $g = 22$, we have an M value of the order of 10^7 nodes. Still no node is away from the boundaries more than 11 bonds. This means that even for the largest possible molecules the boundary effects will always be present. If we were to simulate diffusion on these structures, then the diffusing particles feel the presence of the boundaries very fast, and their behavior is very different than if the system were infinite. This was clearly shown earlier [13], where in systems of several million molecules with $g = 22$ the boundary effects were manifested in as few as 20 or 50 Monte Carlo steps, thus making it impossible to directly monitor any random walk property. Revisitation of the same sites occurred at very early times, while the diffusing particles were all gathered at the boundaries of the structure. Using cyclic boundary conditions (which in normal lattices is a useful concept) did not offer here any noticeable improvements because of the high degree of asymmetry of the structure. Also, using other sets of boundary conditions, as described in Ref. [13] did not solve this problem. Clearly then, one would need to devise a method by which an infinite size system would be used. We, therefore, develop here two algorithms that perform diffusion on a dendrimer without defining its entire structure. Only the path traced by the diffusing particle is defined, thus making it possible to follow the process as if the system were infinite, or practically extremely large, and thus we avoid the boundary effects. The first algorithm is a "string-type" algorithm, and this class is described in detail in the classic book of Knuth [14]. It simply evaluates the distance (the mean square displacement) between the initial and final points by keeping track of two arrays, first the position array at the beginning of the walk, and then the dynamical array which changes only its last element in each step. Using this algorithm we can also evaluate the number of sites visited, by defining the exact position of the moving particle not by a number, or index, but by a "string" of characters. Here we use 0 and 1 as string elements. We can generate walks on lattices that would be equivalent to generation order $g = 10^4$–10^5, which is large enough for the calculations that we are interested. Finally, the second algorithm is the parent–child labeling (PCL) algorithm. By using this algorithm there is no limit in the size of the system to be simulated (the only limit is the memory of your computer). The PCL algorithm uses a unique labeling method for the node that is visited and its neighboring nodes, i.e. its parent node and its children nodes. The labels change in a way that is equivalent to the spirit of the cluster multiple labeling technique of the percolation problem [15], even though the two problems are unrelated and the implementation is done totally differently in the two cases. The common characteristic that they both have is that we avoid sweeping the entire system structure (which is time and memory

consuming). We use instead dynamical arrays in which we simply manipulate the index values of the index location of the array elements. We will give numericalexamples in the following section where these ideas are fully clarified.

While for some of the properties above there already exist analytical formalisms, and one might question the need of computer simulations, we must stress here that the two algorithms developed here are general enough that they could be very useful for the simulation of different dynamical properties, such as, for example, chemical reactions on dendrimers, etc., for which there is no analytical formalism. At this early stage our goal was to make these calculations possible, surpass the difficulty of finite size effects, and test the results with the existing formalisms.

2. The algorithms

We describe here in detail two algorithms. First, the string algorithm (1) for the calculation of the distance $\langle R\rangle$ traversed on a dendrimer, and also for the number of visited sites, $\langle S_N\rangle$, and for trapping. Then we give the PCL algorithm (2).

2.1. The string algorithm

The algorithm for calculating the $\langle R\rangle$ distance is the following: We assume that the starting point is some random point in the structure, where a particle is placed. Before choosing the random site we first choose its g. Thus we randomly choose a g value, and assume that the starting point will be a site with this chosen g. The value of g of this point is known and recorded. Next we build a one-dimensional array with g number of elements, which take random values in the interval 1 to k, where $k = z - 1$. This array gives the exact position of the starting point in the structure, and it is kept intact for the entire duration of the run. Then the random walk is started, and the particle moves to one of the nearest neighbor sites only, which all have the same probability. A second one-dimensional array is now constructed, which at this point is identical to the first one. When a step is made at random, during the random walk, the second array changes by adding or subtracting one row, according to whether g increases by 1 or decreases by 1, respectively. Each time the move is made towards the root of the dendrimer the size of the array decreases by one, while there is no change to the remaining elements. This is because we keep track only of the exact current position at all times. Each time the move is away from the root (g increases) a new element is added at the end of the array with the use of a random number to set the coordinate for the newly found node. We call it "newly found" because once the particle moves away from it we do not keep any other information about it. Only these two arrays are monitored, the first one at time $t = 0$, and the second one, which dynamically changes during the walk. At the end of the process, we compare the two arrays. We will notice that the top parts in both arrays are identical, while their bottom parts are different. In each array of the two that we have, we subtract the common top part, and finally R will be the sum of the remaining two bottom parts. For example, if the 2 arrays have sizes of 15, and 100, respectively, and the top 10 rows are identical in both arrays, then $R = (15 - 10) + (100 - 10) = 95$.

Next, we calculate the S_N property. Again, as in both algorithms the entire structure is not generated. Here we generate only the single path leading to the initially randomly chosen point of origin. For coordination number $z = 3$ each node has two children. We will use 0 and 1 for the identification of the left and right children, respectively. The first node of the dendrimer (the core) is assigned an index value of 1. The following nodes in the ensuing generations are randomly chosen. Lets say that initially the generation happens to be $g = 20$. An example of a node on this g could be the string 10011011010110110111. We also know that the parent of this node is the string 1001101101011-011011, the left child of it is the string 100110110101101101110, and the right child of it is the string 100110110101101101111. All these quantities are defined as character strings in the codes and not integer numbers. Finally, we create an array on which the first element is this 20-character string. A random number is chosen to decide to which adjacent node the particle will move to. Once this is done, the string for the new

node is formed, which in this case is either the 19-character parent string or one of the 21-chracter children strings. This is now the second element in the array. In the same fashion we choose all array elements. The number of the array elements is equal to the number of steps. Every time we form a new string (i.e. in every step of the random walk) we compare this string with all previous strings. If there is at least one string that is the same with the current string, then this means that the new site was already visited previously. If there is no string in the entire array equal to the current string, then this site is a new one, and S_N is increased by one.

2.2. The Parent–child labeling algorithm

As we mentioned previously using this algorithm it is possible to simulate random walks on dendrimers of unlimited number of generations. The idea is the same in both algorithms, i.e. that we do not keep the entire structure in memory, but only the tracing of the path of the diffusing particle. In order to do this we define a number of arrays equal to z, the coordination number. We will describe one example in detail, for which we will use $z = 3$, meaning that each node has three neighboring nodes, and thus we need three arrays, say, **P** (for parent), and **C1** and **C2** (for the two children). We associate **C1** to be the one that leads to the left child of the two children and **C2** the one that leads to the right one. The size of the arrays must be equal to the total number of steps to avoid the array overflow in case of possible straight ballistic motion. Thus, for a walk of one million steps we need three arrays, each of one million elements. We define an index N, which follows the random walk step-by-step and assigns values to the nodes, i.e. the indices of the parent and the child for each step. The position on the trajectory is thus given by this index, and it is the same in all three arrays.

We initialize the arrays by assigning the value of -1 to all array elements in all three arrays in order to indicate that there are no visited connections between nodes of the tree. Up to this point nothing is associated with the trajectory. Every time that the particle is on a node and a new step is to be taken we start this by drawing a random number to decide to which adjacent node the new step will be taken, a common procedure for all types of random walks. As the random walk starts N is still undefined. The first value that N will take is arbitrarily the value $N = 1$, we give this value to the index of the site of the origin of the random walk. We also define $N_{\max}$ as the maximum value that N has taken in the entire calculation. Initially $N_{\max} = N = 1$. For the first step and for all subsequent steps the first question that is asked is to find out if the new site has been visited before or not. We check on the index of the site, and if the index is -1 then the site has not been visited. If it is anything else other than -1 then it has been visited before. If it is an unvisited site then we increase by one the value of a counter that keeps track of S_N. If the site has been visited before this counter stays the same. The random walk proceeds as follows, as shown on Fig. 2: Say that we are in the middle of a run and it happens that the index of a particular site is $N = 10$, while the value of $N_{\max}$ is $N_{\max} = 13$. A new step is to be taken, and the random number drawn leads us to the left child. Let's consider the indexing in the two possibilities separately. We check on $\mathbf{C1}(10)$, and find out that $\mathbf{C1}(10) = -1$. This is a new site, and thus we place the value $N_{\max} + 1 = 14$ as the index of $\mathbf{C1}(10)$, i.e. $\mathbf{C1}(10) = 14$. At the same time we place the value 10 at the parent of 14, i.e. $\mathbf{P}(14) = 10$. This is shown in Fig. 2A. For each step it is necessary to check and do both values of the two indices.

Now let us consider the second possibility that the new site has been visited before. We check on

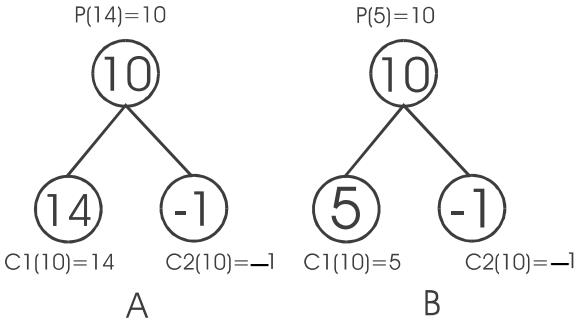

Fig. 2. Example of PCL for two cases: (A) the new site has not been visited before and (B) the new site has been visited before the current step.

the index of the new site and find that it has the value of 5, i.e. $\mathbf{C1}(10) = 5$. We do not change the values of the indices at this point, but we change the N value to $N = 5$. Thus we still have $\mathbf{C1}(10) = 5$, and $\mathbf{P}(5) = 10$. See Fig. 2B for details.

The same algorithm is used for the simulation of the trapping problem. This is now done slightly different than the traditional method. Initially no traps are placed on the structure. When the diffusing particle makes a step to another node we check to see if this node has been visited before or not. If it has been visited before then we know that this site is not a trap site, and the walk continues normally. If it has not been visited before, then this site is checked to see if it is a trap site. We do this by drawing a random number. If this number is less than the trap concentration C, then the node is a trap, and the random walk stops immediately. In the opposite case it is simply another regular node, we label it as visited, and the walk continues.

3. Results

In Fig. 3 we show the behavior of R as a function of time for several different z values. The data shown are results of simulations using algorithm 1. Here R is considered to be the excursion distance from the point of origin, and it is measured in terms of structure bonds traversed during the random walk. This distance R is the shortest distance between these two points (original and final points) on the Cayley tree. While during the random walk a bond can be traversed many times, when the distance R is calculated no bond is traversed more than once, in order to find the shortest possible distance.

We observe that R grows linearly with time, for any z. Accordingly, one could expect that the mean square displacement, R^2 would grow here with N^2 This is very different from what we have in lattice systems, where, say for a one-dimensional lattice, $\langle R\rangle = 0$, and $\langle R^2\rangle = N$. This different result is expected because of the nature of the dendrimer structure. The probability of return to the origin is extremely small, because as we discussed earlier, the walk is an effective biased walk pointing away from the point of origin. For longer times this probability goes to zero. The mean square displacement is shown in Fig. 4 plotted in log–log form. We observe that indeed there is a slope of 2 for any branching ratio z. We conclude that diffusion in dendrimers is ballistic in nature, rather than stochastic, as it is on normal lattices.

The exact prediction for the $\langle R\rangle$ quantity is given by:

$$\langle R\rangle = \frac{z-2}{z}N. \tag{4}$$

This is rationalized as follows: Since the motion is ballistic in nature, the probability of a transition

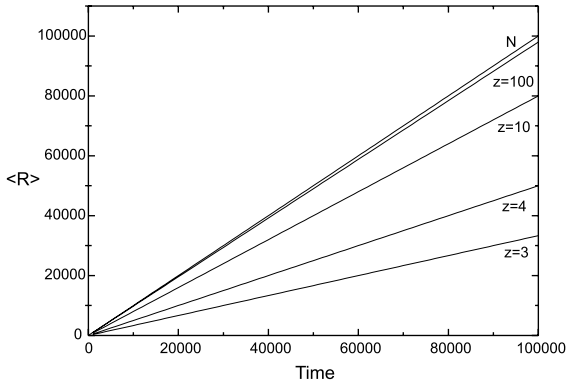

Fig. 3. The average distance $\langle R\rangle$ as a function of time, for random walks on dendrimer structures, for several different z values. The top straight line is $\langle R\rangle = x$, i.e. a straight line with a slope of $\pi/4$. The generation order is $g = 1\,000\,000$.

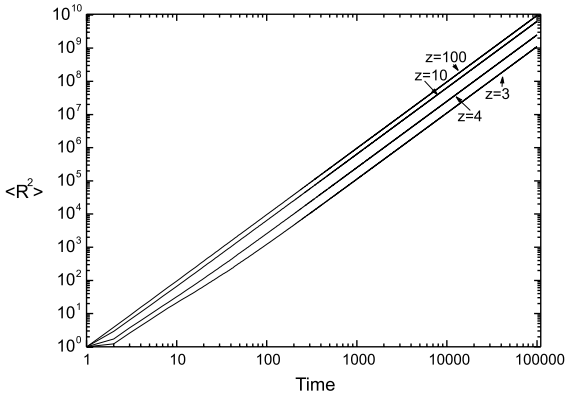

Fig. 4. Log–log plot of the mean square displacement, $\langle R^2\rangle$, as a function of time, for random walks on dendrimer structures for several different z values.

towards the core is balanced by a similar probability along one of the bonds towards the periphery. The rest of the bonds towards the periphery contribute to the ballistic motion. Thus, from a total of z bonds one must subtract the contribution of two bonds, and the fraction $(z-2)/z$ is the one remaining for the ballistic motion. Eq. (4) is in full agreement with the data of Fig. 3, as it can be easily seen.

Since the stochastic walk on a dendrimer is effectively a biased walk, it is of interest to consider a model on which this inherent characteristic of dendrmers is taken out so that effectively this bias is negated. This can simply be accomplished by assigning a $1/2$ probability for the particle to go up towards the core along the single bond, and also $1/2$ together to the $(z-1)$ bonds to go down towards the periphery. If $z=3$, then the latter probability would be $1/4$ along each bond. We performed such calculations, and the results are given in Fig. 5. Here we give the mean square displacement, $\langle R^2\rangle$, as a function of time for several different z values. All z values produce exactly the same result, and we observe that all data collapse to the same straight line. The slope of the line is 1, meaning that this model reproduces exactly the diffusion model on regular lattice systems. One could surmise that this observation is reasonable, because the $\langle R^2 \propto N$ diffusion law holds in any dimensionality, and one would expect it to also hold here. Also not surprising is the fact that this observation is independent of z, because in this model the direction of motion, for steps away from the core, has the same equal probability for any z value.

In Fig. 6 we show the results for the calculation of S_N. In the same graph we also give the theoretical predictions for the regular one-, two-, and three-dimensional lattices (as full lines, no symbols). For these lattices the well known results are that for one-dimensional systems $S_N \propto N^{1/2}$ for two-dimensional $S_N \propto N/\log N$, and for three-dimensional $S_N \propto N$. The theoretical asymptotic prediction for S_N on the dendrimers is given by Eq. (1). We show the cases of $z=3$ and $z=4$, where the line is the theoretical prediction and the symbols on the lines are the results of simulations. We used both algorithms 1 and 2 and received identical data. We have very good agreement between all these methods. In contrast, in the past [13], where finite size dendrimers were used, there was no agreement because the finite size effects dominate even at very early times. We conclude that for the S_N property dendrimers resemble the three-dimensional lattice behavior, differing only in the prefactor, with the case of $z=3$ being less efficient, while for $z>3$ dendrimers are always more efficient, the efficiency increasing with z.

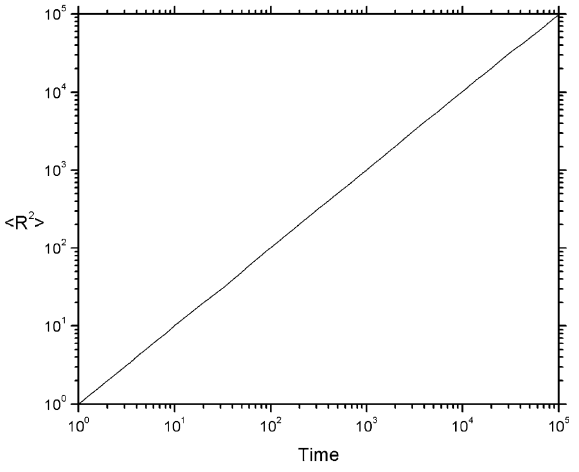

Fig. 5. The mean square displacement, $\langle R^2\rangle$, as a function of time, for random walks on dendrimer structures, with the "no-bias" model of transitions. The calculation is for the case of $z=3$, but as discussed in the text, all z values produce exactly the same result.

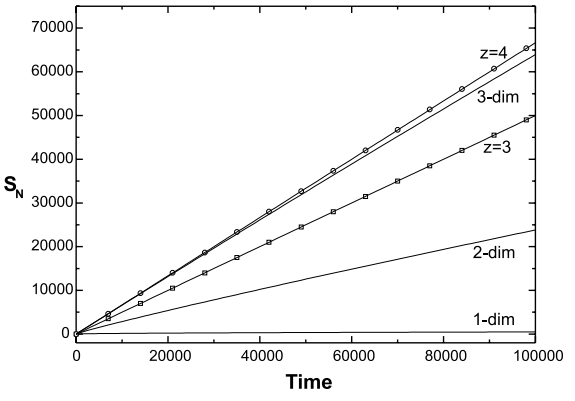

Fig. 6. S_N as a function of time for regular diffusion on dendrimers with $z=3$ and $z=4$. Also shown are the cases of diffusion on the three regular lattice dimensionalities for comparison purposes.

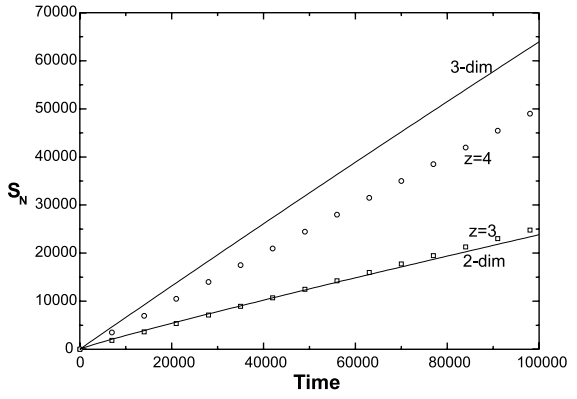

Fig. 7. S_N as a function of time for random walks on dendrimers with the "unbiased" model of transitions. Shown are the cases of $z = 3$ and $z = 4$. In addition we give again the cases of diffusion on the three regular lattice dimensionalities for comparison purposes.

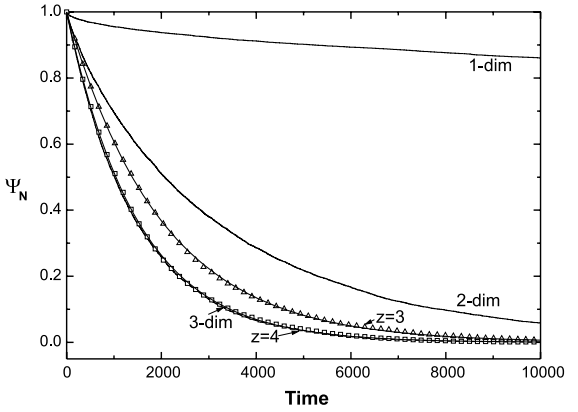

Fig. 8. The survival probability, Ψ_N, as a function of time for regular random walk diffusion on the dendrimer structures. The concentration of traps is $C = 10^{-3}$. Two cases are shown, for $z = 3$ and $z = 4$. Also, for comparison, we show the simulation results for regular one-, two-, and three-dimensional lattices. The simulation data are the symbols, while the curves are the results of Eq. (2).

We then calculated the S_N quantity using the "no-bias" model, and the results are shown in Fig. 7. As we see by comparing with Fig. 6 the "bias" model shows a higher S_N behavior rather than the unbiased case. This is true for every biased walk on any lattice.

Finally, a property that is also based on the probability for return to the origin and it is in the same class of random walk properties is that of trapping. The survival probability Ψ_N is definitely a decaying function of time, but it is far from being a simple exponential. While its functional form is very complicated even on regular lattices [11], we resort here to the Rosenstock approximation and find approximate solutions which are quite satisfactory. In Figs. 8 and 9 we show the survival probability as a function of time for two z values, $z = 3$ and $z = 4$, both for the case of a symmetric walk and an unbiased walk. The symbols are results from direct simulations, while the full lines are the solutions of Eq. (2). We observe an excellent agreement between the two. Additionally we find that diffusing particles survive longer in the case of no bias. This is a new finding, and apparently it happens because the bias characteristic "pushes" particles faster towards the traps, while in the unbiased case we have the well known effect that particles are found in large size trap-free regions in which they spend relatively long periods

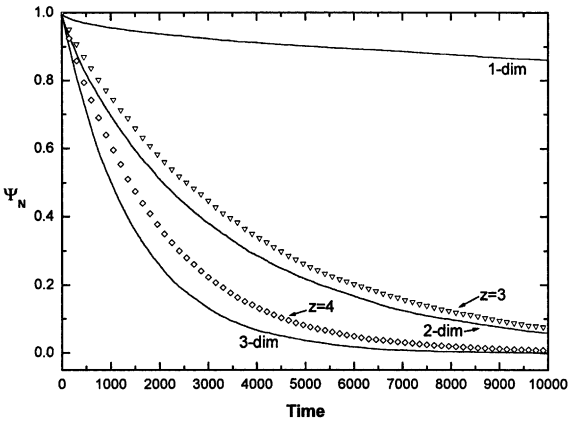

Fig. 9. Similar plot for Ψ_N, as in Fig. 8, but here we use a random walk model with the "no-bias" characteristic in the diffusional motion.

of time without trapping. Of course, this trend is in parallel with the S_N trend. The present results, in this sense, serve also as a verification of the Rosenstock approach applied to non-lattice systems.

4. Discussion and conclusions

Summarizing, we have performed computer simulations on dendrimer structures, modeled by

geometrical Cayley trees, and investigated in detail well-known diffusion properties, especially the ones that have been heavily studied in the past on discrete lattices. While the results are rather simply derived, the numerical simulations are useful because they point to the peculiar characteristic of motion on these structures. We point out the character of the "bias" of the random walk towards the periphery of the system, something that is shown in the results of the walk excursion, R, and the number of sited visited, S_N. We show that regular diffusional motion is recovered only when the bias factor is taken out. Because of the finite size effects which make computer simulations practically impossible in entire dendrimer systems, we develop two algorithms for all diffusion properties, which are size independent and thus, we can avoid the finite size effects. These algorithms are based on string character definitions, and on a new method of labeling the indices of the nodes. We thus recover the exact analytical solution of Hughes–Sahimi for the S_N property. In the past [13] this was impossible to achieve because of finite size effects, which were manifested very strongly after only a very small number of steps, and thus no agreement with the analytical forms was possible.

Acknowledgements

This work was supported by NATO grant SfP971940, by the Greek-Russian bilateral grant GGET9870/2000, and by RFBR grants 99-02-17326, 00-15–96707 and 00-02-16601a. We would like to thank Mr. Hailin Peng for useful discussions and suggestions, and Mr. Kosmas Kosmidis for producing Fig. 5.

References

[1] D.A. Tomalia, A.M. Naylor, W.A. Goddard, Angew. Chem. Int. Ed. Engl. 29 (1990) 138.
[2] M.R. Shortreed, S.F. Swallen, Z.Y. Shi, W. Tan, Z. Xu, C. Devadoss, J.S. Moore, R. Kopelman, J. Phys. Chem. 101 (1997) 6318.
[3] S. Tretiak, V. Chernyak, S. Mukamel, J. Phys. Chem. 102 (1998) 3310.
[4] A. Bar-Haim, J. Klafter, R. Kopelman, J. Am. Chem. Soc. 119 (1997) 6197.
[5] S.F. Swallen, R. Kopelman, J.S. Moore, C. Devadoss, J. Molec. Struct. 485–486 (1999) 585.
[6] S.F. Swallen, M.R. Shortreed, Z.Y. Shi, W. Tan, Z. Xu, C. Devadoss, J.S. Moore, R. Kopelman, in: P.N. Prasad (Ed.), Dendrimeric Antenna Supermolecules with Multistep Directed Energy Transfer, Plenum Press, New York, 1998.
[7] J.S. Lindsey, J. Am. Chem. Soc. 116 (1994) 9759.
[8] C.A. Bignozzi, R. Argazzi, J.R. Schoonover, G.J. Meyer, F. Scandola, Sol. Energy Sol. Cells 38 (1995) 187.
[9] B.D. Hughes, M. Sahimi, J. Stat. Phys. 29 (1982) 781.
[10] G.H. Weiss, Aspects and Applications of the Random Walk, North-Holland, Amsterdam, 1994.
[11] M.D. Donsker, S.R.S. Varadhan, Commun. Pure Appl. Math. 28 (1975) 525;
Commun. Pure Appl. Math. 32 (1979) 721.
[12] H.B. Rosenstock, Phys. Rev. 187 (1969) 1166;
J. Math. Phys. 11 (1970) 487.
[13] P. Argyrakis, R. Kopelman, Chem. Phys. 261 (2000) 391.
[14] D.E. Knuth, The Art Of Computer Programming: Fundamental Algorithms, third ed., Addison-Wesley, Reading Mass, 1997.
[15] J. Hoshen, R. Kopelman, Phys. Rev. B 14 (1976) 3428.