Design and analysis of a bio-inspired search algorithm for peer to peer networks *

Niloy Ganguly, Lutz Brusch, Andreas Deutsch

Center for High Performance Computing, Dresden University of Technology, Dresden, Germany. {niloy, brusch, deutsch}@zhr.tu-dresden.de

Abstract. Decentralized peer to peer ($p2p$) networks like Gnutella are attractive for certain applications because they require no centralized directories and no precise control over network topology or data placement. The greatest advantage is the robustness provided by them. However, flooding-based query algorithms used by the networks produce enormous amounts of traffic and substantially slow down the system. Recently, flooding has been replaced by more efficient k-random walkers and different variants of such algorithms. In this paper, we report immune-inspired algorithms for searching peer to peer networks. The algorithms use the immune-inspired mechanism of affinity-governed proliferation to spread query message packets in the network. Through a series of experiments, we compare the proliferation mechanism with different variants of random walk algorithms. The detailed experimental results show message packets undergoing proliferation spread much faster in the network and consequently proliferation algorithms produce better search output in $p2p$ networks than random walk algorithms. Moreover, theoretical results by calculating the packet spreading speeds are reported which provide an understanding of the improved performance of the proliferation based search algorithm.

1 Introduction

Among different desirable qualities of a search algorithm for peer to peer ($p2p$) networks, robustness is a very important aspect. That is, the performance of a search algorithm should not radically deteriorate in face of the dynamically changing condition of the network. As is known, the big share of Internet users, consequently participants in $p2p$ networks, still use slow and unreliable dial-up modems and also leave the community at very short intervals. Thus in order to give robustness a high priority, precise routing algorithms for forwarding query message packets are generally avoided. Instead random forwarding of the message packets is preferred [9]. The goal of this paper is to study more efficient alternatives to the existing k-random walk. In this connection, we draw our inspiration from the immune system.

* This work was partially supported by the Future & Emerging Technologies unit of the European Commission through Project BISON (IST-2001-38923).

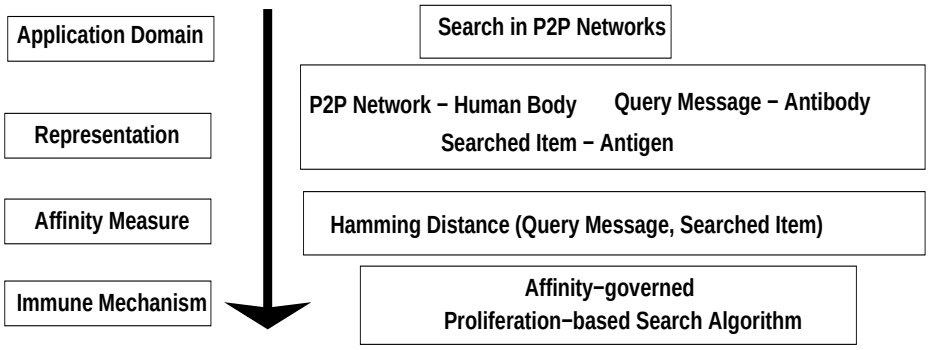


Fig. 1. Immune system concepts used to develop search algorithms

Our algorithm has been inspired by the simple and well known mechanism of the humoral immune system where B cells upon stimulation by a foreign agent (antigen) undergo proliferation generating antibodies. Proliferation helps in increasing the number of antibodies while mutation implies a variety of generated antibodies. Consequently the antibodies can efficiently track down the antigens (foreign bodies). *Fig. 1* provides an illustration explaining how we have mapped immune system concepts to our search problem. In our problem, the query message packet is conceived as antibody which is generated by the node initiating a search whereas antigens are the searched items hosted by other constituent members (nodes) of the $p2p$ network. Like in the natural immune system, the packets undergo proliferation based upon the affinity measure between the message packets and the contents of the node visited which results in an efficient search mechanism. The work presented here is further development of the works reported in [1, 2, 3, 4].

In the next section, we detail the modeling abstractions upon which the algorithms are based. Moreover, we elaborate our algorithms as well as different variants of k-random walk algorithms. Furthermore, the evaluation metrics used to compare the different schemes is introduced. The experimental results are noted next in *Section 3*. *Section 4* provides a theoretical outline explaining the rationale behind the experimental results.

2 Modeling and Evaluation Methodology

It is impossible to model the complete dynamics of a $p2p$ system. In this paper, we do not attempt to resolve small quantitative disparities between k-random walk and proliferation algorithms, but instead are trying to reveal fundamental qualitative differences. While our simple models do not capture all aspects of reality, we hope they capture the essential features needed to understand the fundamental qualitative differences between k-random walk and proliferation algorithms.

2.1 Model Definition

$P2p$ networks are networks formed through associations of computers, each providing equivalent services, eg. search facility, to the network. Thus, each peer can be conceived as both client and server of a particular service [9]. To model a search service, we focus on the two most important aspects of a $p2p$ system: $p2p$ network topology, query and data distribution. For simplicity, we assume the topology and distribution do not change during the simulation of our algorithms. For the purpose of our study, if one assumes that the time to complete a search is short compared to the time of change in network topology and change in query distribution, results obtained from the fixed settings are indicative of performance in real systems.

Network Topology : By network topology, we mean the graph formed by the $p2p$ overlay network; each $p2p$ member has a certain number of neighbors and the set of neighbor connections forms the $p2p$ overlay network. For our studies, we use random graphs. Random graphs are considered as the best type of topology to represent the majority of the realistic network topologies formed in the Internet [5, 9]. In the experiments reported in this paper, we have considered different random graphs each having 10000 nodes. However, their mean node indegree differs. The random graphs have been generated with the help of the topology generator BRITE[7]. In *Fig. 2*, we show the node degree distribution followed by one of the representative graphs. This particular graph has 10000 nodes and mean node indegree $\mu_{nd} = 4$.

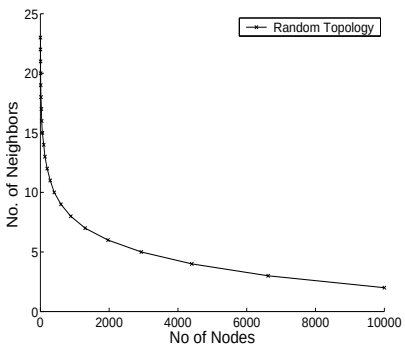


Fig. 2. Cumulative distribution of node degrees in Random graph, 10000 nodes, with $\mu_{nd} = 4$.

Data and query distribution : Files are conceived as conglomeration of keywords [6]. Hence the data distribution is represented in terms of keywords. It is assumed that there are 2000 different keywords in the system. Each node hosts some keywords. The number of keywords (not unique) in each node follows a Poisson distribution with mean $\mu_{kw} = 1000$. The data profile (D) of each node can therefore be represented as

$$D =< (\delta_1, n_1), (\delta_2, n_2), \cdots > \ \& \ n_1 + n_2 + \cdots = N$$

where δ_i are each individual keywords and n_i indicates their weights (number of times they are present in the node), N represents the total number of keywords.

The query comprises of single or multiple keywords. The query (M) can be represented as

$$M =< m_1, m_2, \cdots >$$

where m_i represents each individual keyword. For 95% of the cases, the query length (say n) is ≤ 5 while it is between 6 to 10 in the rest 5% case. In the 95% cases where the query length is ≤ 5, each length 1 to 5 is equiprobable. This is similar for the rest 5% case.

Zipf's distribution[11], is chosen to distribute each of the 2000 unique keywords in the network. In Zipf's distribution the frequency of occurrence of some event (here keywords) t, as a function of the rank r, where the rank is determined by the above frequency of occurrence, follows a power-law $t_i \propto \frac{1}{r^a}$. In the experimental setup, Zipf's exponent (value of a) for both the query and the data is 1. The ranking of keywords in terms of frequency is the same for both data and query distribution. For instance, the most popular query keyword is also the most popular data keyword.

We now describe the proliferation and random walk algorithms.

2.2 Algorithms

In this section, we introduce two proliferation based as well as two random walk based search algorithms. The important aspects of all these algorithms are that although random walk or proliferation is exhibited by the message packets, the algorithms are independently implemented by each node. And coordinated behavior of the nodes produces the required packet dynamics. All the algorithms can be expressed in terms of the following basic premise.

Basic Premise : The search in our *p2p* network is initiated from the user peer (U). The user peer (U) emanates k ($k \geq 1$) message packets (M) to its neighbors - the packets are thereby forwarded to the surroundings. In the following we present the search initiation process in algorithmic form.

Algorithm 1 InitiateSearch(U)
Input : *Signal to initiate search.*
Form Message Packet (M) $= < m_1, m_2 \cdots >$; m_i *represents each individual keyword*
Flood k *message packets*(M) *to the neighbors of the user peer.*

The message packets travel through the network and when a node (say A) receives a message packet (M), it performs the following two functions.
Function 1 :- It checks whether any $\delta_i \in D$ of A is equal to any $m_i \in M$ (incoming message). The number of successful matches Sm is represented by the following equation.

$$Sm = \sum_{i=1}^{N} \sum_{j=1}^{n} (m_j \oplus \delta_i) \times n_i \tag{1}$$

where $m_j \oplus \delta_i = 1$, if $m_j = \delta_i$, else 0; N is the total number of keywords present in the node while n represents the length of the search query. The node reports the number of successful matches - Sm.
Function 2 :- It forwards the content of the message packet in some *defined* manner to its neighbor(s).

In algorithmic form, we can represent the functions as *Reaction_p2p*:

Algorithm 2 Reaction_p2p(A)
Input : Message packet(M)
*Calculate Sm from D & M /*Function 1*/*
Algorithm Message_Forward(A) / Function 2*/*

Each of the proliferation and random walk schemes defines *Algorithm Message_Forward(A)* differently. Elaboration of the algorithms corresponding to each of the schemes follows.

Proliferation $\mathcal{P}$: In the proliferation scheme, the packets undergo proliferation at each node they visit. The proliferation is guided by a special function, whereby a message packet visiting a node proliferates to form N_p message packets which are thereby forwarded to the neighbors of the node.

Algorithm 3 $\mathcal{P}$(A)
Input : Message packet(M)
Produce N_p message packets(M)
Spread the N_p packets to N_p randomly selected neighbors of A

The function determining the value of 'N_p' ensures that N_p is $< \eta(A)$, where $\eta(A)$ is the number of neighbors of A and ≥ 1. [Note that if $N_p = 1$, proliferation behaves similar to random walk.]

Restricted Proliferation ($\mathcal{RP}$) : The restricted proliferation algorithm, similar to $\mathcal{P}$, produces N_p messages. But these N_p messages are forwarded only if the node A has $\geq N_p$ free neighbors. By 'free', we mean that the respective neighbors haven't been previously visited by message M. If A has $\mathcal{Z}$ 'free' neighbors, where $\mathcal{Z} < N_p$, then only $\mathcal{Z}$ messages are forwarded, while the rest are destroyed. However, if $\mathcal{Z} = 0$, then one message is forwarded to a randomly selected neighbor. The rationale behind the restricted movement is to minimize the amount of message wastage. Because, two packets of message M visiting the same peer essentially means wastage of the second packet.

Algorithm 4 $\mathcal{RP}$(A)
Input : Message packet(M)
Produce N_p message packets (M)
$\mathcal{Z}$ = No of 'free' neighbors
if ($\mathcal{Z} \geq N_p$)
    Spread the N_p packets in N_p randomly selected neighbors of A
else
    if ($\mathcal{Z} > 0$)
        Spread $\mathcal{Z}$ packets in $\mathcal{Z}$ free neighbors of A
        Discard the remaining (N_p - $\mathcal{Z}$) packets
    else
        Forward one message packet to a randomly selected neighbor of A
        Discard the remaining (N_p - 1) packets

We now elaborate the function which controls the amount of proliferation.

Proliferation Controlling Function : The proliferation of message packets at any node A is heavily dependent on the similarity between the message packet (M) and the data profile (D) of A. In this connection, we define the measure

of similarity between the data profile (D) of the node and the message packet (M).

$$Sim = \frac{Sm}{N}$$

where the value of Sm is calculated through Eq. (1). [Note $Sm \leq$ N, so the value of $Sim \leq 1$.] The number of packets N_p proliferated is defined on the basis of Sm in the following manner

$$N_p = 1 + Sim \times (\eta - 1) \times \rho$$

where η represents the number of neighbors the particular node has; ρ represents the proliferation constant, it is ≤ 1. (ρ is set to 0.5 in all our experiments.) The above formula ensures that $1 < N_p \leq N$.

We now describe a simple k-random walk algorithm and subsequently two different variations of it.

k-random walk ($\mathcal{RW}$) : In k-random walk, when a peer receives a message packet after performing the task of comparison, as mentioned in *Algo. 2*, it forwards the packet to a randomly selected neighbor. The algorithm ($\mathcal{RW}$) is quite straightforward and is defined as

Algorithm 5 $\mathcal{RW}$(A)
Input : Message packet(M)
Send the packet M to a randomly chosen neighbor peer

The restricted random walk ($\mathcal{RRW}$) algorithm which is similar to $\mathcal{RP}$ (*Algo. 4*), is discussed next.

Restricted Random Walk ($\mathcal{RRW}$) : In $\mathcal{RRW}$, instead of passing the message (M) to any random neighbor, we pass on the message to any randomly selected 'free' neighbor. However, if there is no 'free' neighbor, we then pass on the message to any randomly selected neighbor.

Algorithm 6 $\mathcal{RRW}$(A)
Input : Message packet(M)
Send the packet M to a randomly chosen 'free' neighbor peer
If (no 'free' neighbor)
    Send the packet M to a randomly chosen neighbor peer

2.3 Metrics

In this paper we focus on efficiency aspects of the algorithms solely, and use the following simple metrics in our abstract $p2p$ networks. These metrics, though simple, reflect the fundamental properties of the algorithms.

(a). *Success rate:* The number of similar items found by the query messages within a given time period.

(b). *Coverage rate:* The amount of time required by the messages to cover a percentage of the network.

(c). *Effectivity per message:* The number of search items produced by a single message .

3 Simulation Results

The experimental results compare the efficiency of different algorithms (Algo. 3 - 6), with respect to the metrics defined in *section 2.3*.

As mentioned earlier, each of the above algorithms is distributed in nature and the nodes perform the task independently of the others. However, to assess the speed and efficiency of the algorithm, we have to ensure some sort of synchronous operation among the peers. In this context we introduce the concept of time whereby it is assumed that in one time unit, all the nodes in the network execute the algorithm once. That is, if a peer has some messages in its message queue, it will process one message within that time frame. We believe although approximate, it is a fair abstraction of reality of *p2p* networks where each node is supposed to provide equivalent services. The sequence of operation of the peers during one time step is arbitrary. The length of the message queue is considered to be infinite.

In order to assess the efficiency of different algorithms, we have also to guarantee fairness of 'power' among them which is explained next.

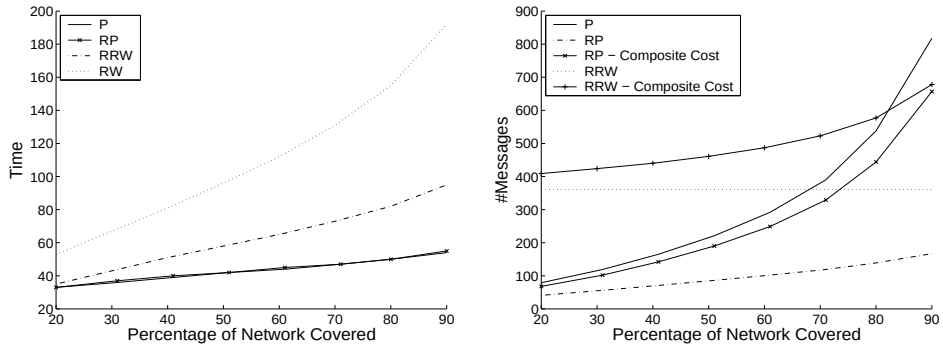
3.1 Fairness in power

To ensure fair comparison among all the processes, we must ensure that each process ($\mathcal{P}$, $\mathcal{RP}$, $\mathcal{RW}$, $\mathcal{RRW}$) participates in the network with the same 'power'. To provide fairness in 'power' for comparison of a proliferation algorithm (say $\mathcal{P}$) and a random algorithm (say $\mathcal{RW}$), we ensure that the total number of query packets used is roughly the same in all the cases. Query packets determine the cost of the search; too many packets cause network clogging bringing down the efficiency of the system as a whole. It can be seen that the number of packets increase in the proliferation algorithms over the generations, while it remains constant in the case of random walk algorithms. Therefore the number of message packets - k in *Algo. 1* is chosen in a fashion so that the aggregate number of packets used by each individual algorithm is roughly the same.

Besides the cost of the message packets, during comparison between a restricted algorithm (say $\mathcal{RRW}$) and a non-restricted algorithm (say $\mathcal{P}$), we also have to keep in mind that checking 'whether a node was earlier visited or not' involves a cost; this also should be taken into consideration when defining 'fairness'. Therefore, the composite cost[2] for a restricted algorithm can be defined as $C_{comp} = X + \alpha \cdot L$, where X is the average number of message packets, L is the number of neighbor lookup, while α is the ratio of cost of lookup to cost of actually sending the message; α normally ≤ 1. However, in this case, since message length is small, we consider the worst case scenario of $\alpha = 1$ to depict our results.

To ensure fairness in 'power' between two proliferation algorithms (say [$\mathcal{P}$ & $\mathcal{RP}$]), we keep the proliferation constant ρ and the value of k the same for both

[2] Henceforth, cost or simple cost indicates the cost of message packets while composite cost always means total cost of messages and neighbor lookup.



a. Time taken to cover the network b. Cost incurred to cover network

Fig. 3. Graphs plotting the cost and network coverage time of $\mathcal{P}$, $\mathcal{RP}$, $\mathcal{RRW}$, $\mathcal{RW}$ algorithms in random network. The proliferation constant considered here is $\rho = 0.5$.

processes. The value of k for the proliferation algorithm is generally set as $k = \eta(U)$, where $\eta(U)$ is the indegree of the initiator peer U.

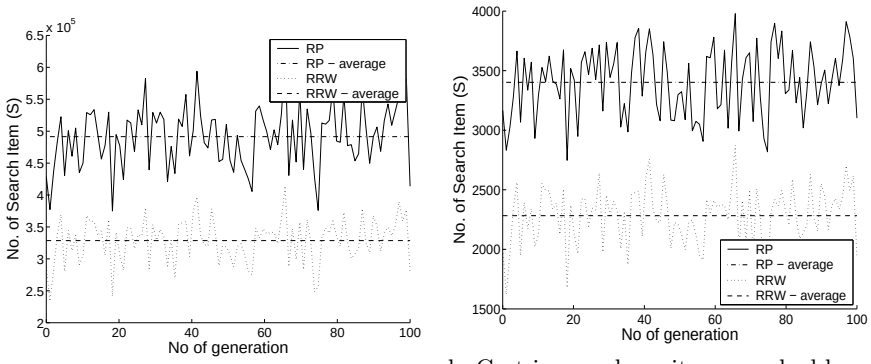
3.2 Experimental Result - Network Coverage

As mentioned in *Sec 2.1*, we use a random graph, to evaluate the time taken by the message packets to visit all the nodes of the network using different forwarding algorithms. The particular random graph considered here has 10000 nodes and mean node indegree $\mu_{nd} = 4$. Its distribution is shown in *Fig. 2*. The experiment, *network coverage* is detailed in the following two paragraphs.

COVERAGE : In this experiment, upon initiation of a search (*Algo. 1*), the search operation (*Algo. 2*) is performed till the message packets cover the entire network. The experiment is repeated 1000 times on randomly selected initial nodes.

During the experiment, we collect different statistic at every 10% of coverage of the network that is, we collect statistic at [20%, 30% $\cdots$ 90%, 100%] of network coverage. Since the message forwarding algorithms (*Algo. 3 -6*) are non-deterministic in nature, message packets find it increasingly difficult to visit the last 10% of the network. This is true for all the different variants of message forwarding algorithms. Consequently, in our results we avoid showing results from the last 10% as it only depicts the aberration arising from the finite size of the network.

Fig. 3(a) shows the network coverage rate of different algorithms $\mathcal{P}$, $\mathcal{RP}$, $\mathcal{RRW}$ and $\mathcal{RW}$. The graph plots the % of network covered in the x-axis, while the time taken to cover the corresponding % of network is plotted on the y-axis. It is seen that $\mathcal{P}$ and $\mathcal{RP}$ take almost identical time to cover up the network. The time taken is, however, much less than that taken by $\mathcal{RRW}$ and $\mathcal{RW}$ respectively. The $\mathcal{RRW}$ is much more efficient than $\mathcal{RW}$. We now assess the cost (both simple and composite) incurred by each algorithm to produce the above mentioned performances.



a. Search efficiency of $\mathcal{RP}$ and $\mathcal{RRW}$

b. Cost incurred per item searched by $\mathcal{RP}$ and $\mathcal{RRW}$

Fig. 4. Graphs showing (a). search efficiency, and (b). cost incurred per item searched by $\mathcal{RP}$ and $\mathcal{RRW}$ in random network.

Fig. 3(b) plots the increase in the average number of message packets present in the network (also referred to as cost) in the y-axis with respect to the percentage of network coverage for $\mathcal{P}$, $\mathcal{RP}$ and $\mathcal{RRW}$. For each $\mathcal{RRW}$ and $\mathcal{RP}$, we show two lines, one for simple and composite cost respectively. Comparing, $\mathcal{RP}$ and $\mathcal{P}$, we see that $\mathcal{RP}$ uses a significantly smaller number of messages (about one-fifth) than $\mathcal{P}$ and achieves the same performance. Even composite cost is significantly lower for $\mathcal{RP}$ (657 for $\mathcal{RP}$ and 818 for $\mathcal{P}$). To ensure the fairness criterion, $\mathcal{RRW}$ initially starts with the number of packets which $\mathcal{RP}$ has used on the average to cover the entire network (361), so it stays constant throughout the experiment; however the composite cost steadily increases. It is the same as for $\mathcal{RP}$ at the 90% coverage ratio. The number of messages $\mathcal{RW}$ uses (not plotted) is 1881. 1881 is the average composite cost incurred by $\mathcal{RRW}$ to cover the entire network.

It is found that $\mathcal{RRW}$ is much more efficient than $\mathcal{RW}$. Similarly, $\mathcal{RP}$ is better than $\mathcal{P}$. So, in our subsequent discussions, we drop $\mathcal{P}$ and $\mathcal{RW}$ and concentrate on a comparison between $\mathcal{RP}$ and $\mathcal{RRW}$.

The next experimental results highlight the search efficiency of $\mathcal{RP}$ and $\mathcal{RRW}$.

3.3 Experimental Results - Search Efficiency

To compare the search efficiency of $\mathcal{RP}$ & $\mathcal{RRW}$, we perform the *time-step* experiment on the random graph for $\mathcal{RP}$ and $\mathcal{RRW}$, each spanning over 100 generations. We use the same random graph as in *Sec 3.2*.

TIME-STEP : In this experiment, upon initiation of a search (*algorithm (1)*), the search operation is performed for $\mathcal{N}$ (= 50) time steps. The number of search items (s) found within 50 time steps from the commencement of the search is calculated. From algorithm (2), we know each visited node returns Sm search items (calculation done through Eq. (1)); s is the summation of Sm

over all visited nodes. The experiment is repeated for one generation where one generation is defined as a sequence of 100 searches. The search output (s) is averaged over one generation (100 different searches), whereby we obtain S, where $S = \frac{\sum_{i=1}^{100} s}{100}$. The value of S is used to draw the graphs explained next. In this experiment, $\mathcal{RRW}$ always performs the experiment with k packets where k is the average number of packets used by $\mathcal{RP}$ over 100 generations.

The graph of *Fig. 4(a)* shows the average value S against generation number for $\mathcal{RP}$ and $\mathcal{RRW}$. The x-axis of the graph shows the generation number while the y-axis represents the average number of search items (S) found in the last 100 searches. In this figure, we see that the search results for both $\mathcal{RP}$ and $\mathcal{RRW}$ show fluctuations. The fluctuations occur due to the difference in the availability of the searched items selected at each generation. However, we see that on the average, search efficiency of $\mathcal{RP}$ is almost 1.5-times higher than that of $\mathcal{RRW}$. (For $\mathcal{RP}$, the number of hits $\approx 5 \times 10^5$, while it is $\approx 3.25 \times 10^5$ for $\mathcal{RRW}$.)

Fig. 4(b) displays the effectivity per messages (metrics defined in *Sec. 2.3*) of each scheme. As expected, the effectivity of message packets for $\mathcal{RP}$ is much higher than $\mathcal{RRW}$ (keeping in mind the fairness criterion we follow to generate experimental results). However, one more important point to be noted is that the standard deviation is particularly small in the case of $\mathcal{RP}$. Considering σ_E/μ_E (Std. of effectivity/mean effectivity), it is 0.1 for $\mathcal{RRW}$ while it is just 0.08 for $\mathcal{RP}$. This is because in $\mathcal{RP}$, the packets are not generated blindly, but are instead regulated by the availability of the searched item. Therefore, if a particular searched item is sparse in the network, $\mathcal{RP}$ produces a lower number of packets and vice versa.

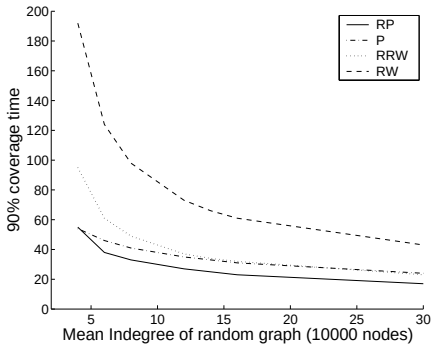


Fig. 5. Graphs illustrating the effect on efficiency with varying indegree

We now describe how efficiency of the different algorithms gets affected when the indegree of the random graph is increased.

3.4 Experimental Results : Changing Indegree

In this experiment, we perform the *coverage* experiment on random graphs each with 10000 nodes, but their mean indegree steadily increases.

Fig. 5 plots the time taken to cover the network (y-axis) by $\mathcal{RP}$, $\mathcal{P}$, $\mathcal{RRW}$ and $\mathcal{RW}$ against different indegree configurations of the random graph (x-axis). The figure shows that as indegree increases each algorithm becomes faster. However, it is to be noted that the speed of random walk algorithms accelerates at a much faster speed. In fact, from around node indegree 12, $\mathcal{RRW}$ and $\mathcal{P}$ take almost identical time. This happens because as indegree of the random graph increases, in effect the dimension (d) of the graph also increases. That is, each

node can reach each other node within a shorter time span. Random walk is particularly smart at higher dimensions which consequently provides the result. However, maintaining high indegree in $p2p$ environment typically is a problem [10]. So, at lower indegree level proliferation algorithms are much more effective than random walk algorithms.

We now summarize the results obtained through the above mentioned experiments

3.5 Summary

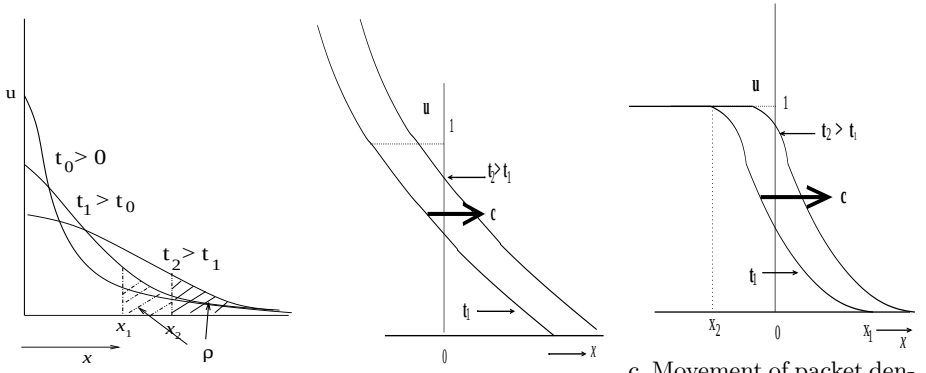
The following is the summarization of the results.
(a). $\mathcal{RP}$ is more effective than $\mathcal{P}$.
(b). $\mathcal{RP}$ is more effective than any random walk algorithm.
(c). $\mathcal{RP}$ has an in-built cost regulatory mechanism.
(d). The search efficiency of $\mathcal{RP}$ is roughly one and a half times higher than $\mathcal{RRW}$.
(e). $\mathcal{RRW}$ becomes more effective as the indegree of the graph increases.

4 Theoretical Justification

In this section, we provide insights into the reasons behind the better performance of proliferation compared to random walk algorithms. We provide our explanation in the framework of a continuum model to derive the macroscopic behavior from knowledge about the individual microscopic behavior of the packets. Unlike the model described in the previous sections, for sake of analysis we assume that the system is synchronous in the sense that all nodes operate at the same time. The network is abstracted as a d-dimensional space where the dimension grows with the average node indegree.

We relate the random walk algorithms to a simple diffusive system where the diffusion starts from the origin. The proliferation algorithms can be conceived as reaction-diffusion systems [8], where besides diffusion of packets, new packets are continuously produced by the existing ones. Each of the two processes (diffusion and reaction-diffusion) spreads the message packets through the network. The insights which we will provide are based upon estimates of the *speed of packet spreading* in the radial direction.

Fig. 6 illustrates the basic features of the two processes. *Fig. 6(a)* refers to diffusion, *Fig. 6(b)* to a reaction-diffusion system with unrestricted proliferation, whereas *Fig. 6(c)* corresponds to a reaction-diffusion system with restricted proliferation. In all three graphs, we plot the density $u(x,t)$ of message packets used to conduct search versus the radial coordinate x. u can be conceived as a normalized measure of the number of packets k. u and k is not quantitatively related in this work, as that is not required to estimate the speed of packet spreading. Each of the three systems is studied in detail below, here we discuss the figures one by one.



a. Spreading of packet density u due to random walk (diffusion).

b. Movement of packet density u due to proliferation (reaction-diffusion 1).

c. Movement of packet density u due to restricted proliferation (reaction-diffusion 2).

Fig. 6. Packet movement with different algorithms in a continuous system with radial coordinate x.

Fig. 6(a) shows three Gaussian curves at three different instances of time (t_0, t_1, t_2, where $t_0 < t_1 < t_2$). As time increases, a particular density of packets (say ρ) travels further away from the center. Moreover, since it follows a Gaussian distribution, at time $t \to 0$, at distance $x \to \infty$, there is some concentration of packets u, where $u \to 0$. However to cover all the nodes at distance x, a finite tangible concentration of packets ρ should reach distance x. As can be seen from the figure, at time t_1, a concentration ρ covers distance x_1 or more while at time t_2 the same concentration has covered $\geq x_2$ distance. Below we calculate the *speed of diffusive packet spreading* for this finite density ρ.

On the other hand, in the case of reaction-diffusion systems (proliferation), the movement of packets follows a traveling front pattern with a uniform front profile. *Figs. 6(b),(c)* show such front profiles at time t_1 and t_2. For example, in *Fig. 6(c)*, we see that at time t_1, till distance x_2, the density $u = 1$, while beyond x_1, it is zero. The second curve at t_2 is just a uniform shift of the first curve, hence characterized by a *front speed*. We now elaborate each of the processes one by one.

A. Random Walk (Diffusion): The random walk has traditionally been modeled as diffusion in continuum systems for which the diffusion equation reads [8]

$$\frac{du}{dt} = D \cdot \frac{d^2u}{dx^2} \tag{2}$$

where D is the diffusion coefficient. We are considering a situation where at time $t = 0$ all packets are concentrated at the origin from where they diffuse outwards. Hence, $u(x, t = 0) = \delta(x{=}0)$, where δ has non-zero value at 0 and 0 otherwise. Therefore, solving the differential equation with the given initial condition, we

obtain

$$u = \frac{1}{(2 \cdot \sqrt{D \cdot \pi \cdot t})^d} \cdot e^{-\frac{x^2}{4 \cdot D \cdot t}} \tag{3}$$

where d is the dimension of the system.

We transform this equation in order to express the position x_ρ of an arbitrarily chosen fixed density $\rho \ll 1$ as a function of time.

$$x_\rho(t) = \sqrt{2 \cdot D \cdot d \cdot t \cdot \log \frac{1}{4 \cdot \rho^{2/d} \cdot D \cdot \pi \cdot t}} \tag{4}$$

The speed c of diffusive packet spreading for any fixed density ρ is obtained by differentiating $x_\rho(t)$ with respect to time.

$$c = \frac{2 \cdot D \cdot d}{2\sqrt{2 \cdot D \cdot d}} \cdot \frac{\log \frac{1}{4 \cdot \rho^{2/d} \cdot D \cdot \pi \cdot t} - 1}{\sqrt{t \cdot \log \frac{1}{4 \cdot \rho^{2/d} \cdot D \cdot \pi \cdot t}}} \tag{5}$$

For most of the time, the logarithm in the numerator is much larger than 1 and we can neglect the 1, hence obtain simplified

$$c = \sqrt{\frac{D \cdot d}{2}} \cdot \sqrt{\frac{1}{t} \log \frac{1}{4 \cdot \rho^{2/d} \cdot D \cdot \pi \cdot t}} \tag{6}$$

The result shows that packet spreading due to random walk becomes faster in higher dimensions as $c \propto \sqrt{d}$ and is slowing down with time as $c \propto \sqrt{\frac{1}{t} \cdot \log \frac{1}{t}}$.

B. Proliferation (Reaction-Diffusion 1): The proliferation algorithm can be modeled as a system which is undergoing diffusion as well as gaining new packets as copies of existing ones at the rate α at each time step. Therefore, the dynamics can be expressed by the following equation

$$\frac{du}{dt} = D \cdot \frac{d^2u}{dx^2} + \alpha \cdot u \tag{7}$$

This equation resembles a variation of a well-studied reaction-diffusion equation, the Fisher equation [8]. We therefore utilize the standard result obtained for the *front speed* in a generalized Fisher equation with reaction term $f(u)$ [8].

$$\frac{du}{dt} = \frac{d^2u}{dx^2} + f(u) \tag{8}$$

has a uniformly moving front as solution and this motion proceeds with the front speed

$$c = 2 \cdot [f'(u_1)]^{\frac{1}{2}}, \tag{9}$$

where $f'(u_1)$ denotes the derivative of $f(u)$ with respect to the packet density u at the position u_1. $u_1 = 0$ is the state of the system that has now yet been visited by the front.

Considering the equation (7), we can rescale it by

$$t^* = \alpha \cdot t; \quad x^* = x \cdot (\frac{\alpha}{D})^{\frac{1}{2}}$$
$$dt^* = \alpha \cdot dt; \quad dx^{*^2} = dx^2 \cdot \frac{\alpha}{D} \tag{10}$$

Therefore, the equation (7) becomes

$$\alpha \cdot [\frac{du}{dt^*} = \frac{d^2u}{dx^{*^2}} + u] \tag{11}$$

Hence $f'(u) = 1$, which implies $f'(u_1) = 1$. Therefore, the front speed c of the system is given by

$$c = \frac{\Delta x}{\Delta t} = \frac{\Delta x^* \cdot \sqrt{\frac{D}{\alpha}}}{\Delta t^* \cdot \frac{1}{\alpha}} = 2 \cdot \sqrt{\alpha \cdot D} \tag{12}$$

This result shows that the speed of packet spreading due to the proliferation algorithm is constant, *i.e.* independent of time. The speed dependents on the proliferation rate α and diffusion constant D but is independent of the dimension d. Hence, the behavior of the proliferation algorithm drastically differs from that of the random walk.

C. Restricted Proliferation (Reaction-Diffusion 2): In the model of restricted proliferation, the number of packets initially increases at a rate α but the packet production rate is lowered as packets encounter more and more packets, *i.e.* density increases. We can conceive the function as logistic population growth model, where $f(u)$ can be modeled by the following equation

$$f(u) = \alpha \cdot u \cdot (1 - u) \tag{13}$$

Therefore, the corresponding reaction-diffusion equation can be written as

$$\frac{du}{dt} = D \cdot \frac{d^2u}{dx^2} + \alpha \cdot u \cdot (1 - u) \tag{14}$$

By using the same rescaling of space and time as above (10), we obtain

$$\alpha \cdot [\frac{du}{dt^*} = \frac{d^2u}{dx^{*^2}} + u(1 - u)] \tag{15}$$

Therefore, $f'(u) = 1$ - 2 u and $u_1 = 0$. Hence $f'(u_1) = 1$ and following the same arguments as in case B. we find $c = 2 \cdot \sqrt{\alpha \cdot D}$. This result implies the same speed of packet spreading and dependence on parameters as in the case of unrestricted proliferation.

To sum up, the above theoretical calculations of speeds of packet spreading due to different algorithms can explain the following observations of the coverage experiments (see *Fig. 3(a)*) and their dependence on the average indegree of the network (see *Fig. 5*).

1. Proliferation algorithms propagate packets faster through the network as their speed is independent of time whereas for random walks packet spreading slows down with time $c \propto \sqrt{\frac{1}{t} \cdot \log \frac{1}{t}}$. This explains the differences between the curves $\mathcal{P}$, $\mathcal{RP}$ and $\mathcal{RW}$, $\mathcal{RRW}$ in *Fig. 3(a)*.
2. Restricted proliferation is as fast as the simple proliferation algorithm, for both the same speed $c = 2 \cdot \sqrt{\alpha \cdot D}$ was calculated. This result is consistent with our finding in *Fig. 3(a)* that the restricted proliferation scheme works as good as the proliferation scheme and both curves $\mathcal{P}$, $\mathcal{RP}$ coincide.
3. Random walk becomes faster as the effective dimension d of the network increases, *i.e.* the indegree increases. This explains the strong dependence of performance on the network indegree for both random walk algorithms in *Fig. 5*.

However, the calculations of the package spreading speeds do not account for the differences in cost effectiveness between the different algorithms.

5 Conclusion

In this paper, we have produced detailed experimental results showing that the simple immune-inspired concept of proliferation can be used to cover the network more effectively than random walk. The proliferation algorithm can regulate the number of packets to be produced during a search operation according to the availability of the searched material, thus improving the efficiency of the search. Moreover, we have provided theoretical results by calculating the packet spreading speeds which explain many of the experimental observations and provide an understanding of the improved performance of the immune-inspired search algorithm.

References

1. N Ganguly, G Canright, and A Deutsch. Design of a Robust Search Algorithm for P2P Networks. In 11^{th} *International Conference on High Performance Computing*, December 2004.
2. N Ganguly, G Canright, and A Deutsch. Design Of An Efficient Search Algorithm For P2P Networks Using Concepts From Natural Immune Systems. In 8^{th} *International Conference on Parallel Problem Solving from Nature*, September 2004.
3. N Ganguly and A Deutsch. Developing Efficient Search Algorithms for P2P Networks Using Proliferation and Mutation. In 3^{rd} *International Conference on Artificial Immune Systems*.
4. N Ganguly and A Deutsch. A Cellular Automata Model for Immune Based Search Algorithm. In 6^{th} *International conference on Cellular Automata for Research and Industry*, October 2004.
5. M. A. Jovanovic, F. S. Annexstein, and K. A. Berman. Scalability Issues in Large Peer-to-peer Networks - A Case Study of Gnutella. Technical Report University of Cincinnati, 2001.

6. Dik L. Lee, Huei Chuang, and Kent Seamons. Document ranking and the vector-space model. *IEEE Softw.*, 14(2):67–75, 1997.
7. A Medina, A Lakhina, I Matta, and J Byers. BRITE: An Approach to Universal Topology Generation. In *Proceedings of the International Workshop on Modeling, Analysis and Simulation of Computer and Telecommunications Systems- MASCOTS*, August 2001.
8. J. D. Murray. *Mathematical Biology*. Springer-Verlag, 1990.
9. A. (Ed) Oram. *Peer-to-Peer: Harnessing the Power of Disruptive Technologies*. O Reilly Books, 2001.
10. G. Pandurangan, P. Raghavan, and E. Upfal. Building low-diameter peer-to-peer networks. *IEEE Journal on Selected Areas in Communications (JSAC)*, 21(6), 2003.
11. G. K. Zipf. *Psycho-Biology of Languages*. Houghton-Mifflin, 1935.