

# Design Of An Efficient Search Algorithm For P2P Networks Using Concepts From Natural Immune Systems<sup>\*</sup>

Niloy Ganguly<sup>a</sup> Geoff Canright<sup>b</sup> Andreas Deutsch<sup>a</sup>

<sup>a</sup>Center for High Performance Computing, Dresden University of Technology,  
Dresden, Germany  
{niloy, deutsch}@zhr.tu-dresden.de

<sup>b</sup>Telenor Research and Development, 1331 Fornebu, Norway  
geoffrey.canright@telenor.com

**Abstract.** In this paper we report a novel and efficient algorithm for searching p2p networks. The algorithm, termed *ImmuneSearch*, draws its basic inspiration from natural immune systems. It is implemented independently by each individual peer participating in the network and is totally decentralized in nature. *ImmuneSearch* avoids query message flooding; instead it uses an immune systems inspired concept of affinity-governed proliferation and mutation for message movement. In addition, a protocol is formulated to change the neighborhoods of the peers based upon their proximity with the queried item. This results in topology evolution of the network whereby similar contents cluster together. The topology evolution coupled with proliferation and mutation help the p2p network to develop ‘memory’, as a result of which the search efficiency of the network improves as more and more individual peers perform search. Moreover, the algorithm is extremely robust and its performance is stable in face of the transient nature of the constituent peers.

## 1 Introduction

Due to their flexibility, reliability and adaptivity, p2p solutions can overcome a lot of disadvantages of traditional client-server systems, and can fit to the dynamically changing Internet environment [1]. This explains the high popularity of file sharing systems like Gnutella, Napster, and Freenet [1]. However, the development of an efficient search algorithm for p2p networks poses a fundamental challenge to researchers. The big share of Internet users still uses dial-up modems, which, besides being slow and unreliable, also leave the community (p2p network) at very short intervals. Therefore, to efficiently manage this totally decentralized system, there is urgent need for developing robust decentralized algorithms, i.e. algorithms which are capable of adjusting to the highly changeable structure of p2p networks.

---

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

The algorithm for search in *p2p* network proposed by us in this paper is termed *ImmuneSearch*. It has been inspired by the simple and well known concept of the humoral immune system where B cells undergo mutation and opportunistic proliferation to generate antibodies which track the antigens (foreign objects). From an information-processing perspective, the immune system is a highly parallel intelligent system. Its general features provide an excellent model of adaptive processes operating at a local level and of useful behavior emerging at the global level. Due to these desirable features of the immune system, there is a growing number of intelligent methodologies inspired by the immune system for solving real-world problems. Examples are: searching for mines [6], anomaly detection in time series data [2], pattern recognition [3], and virus detection [4]. However, to the best of our knowledge, immune system concepts have not been used for designing search algorithms in p2p networks.

*ImmuneSearch* uses affinity-governed proliferation and mutation to spread query message packets (antibodies) across the network. It also evolves the topology of the p2p network in terms of adjusting the neighborhood of the participating peers (information in peers - antigens). This gives rise to a loosely structured network where the overlay topology [1] roughly corresponds to the content in the network. Consequently, the algorithm ensures better quality of service (in terms of the number of search items found within a specified number of steps), and greater efficiency (in terms of the network congestion arising from the query packets) compared to the conventional schemes of random walk and message flooding [5]. The algorithm ensures robustness, that is, stability of performance in face of the transient nature of the network. It also guarantees autonomy to the users, who are not required to store any replicated files on their own machine.

The next section describes the *ImmuneSearch(IS)* algorithm in detail. For modeling purposes, the overlay network [1] responsible for maintaining connections between the peers is represented as a 2-dimensional regular grid topology. *Section 3* details the different simulations performed based upon the algorithm *ImmuneSearch*.

## 2 Simulation Model

This section is divided into two parts. In the first part (Section 2.1) we describe the framework chosen to model the p2p environment. In the second part (Section 2.2) we describe the *ImmuneSearch* algorithm.

### 2.1 Environment Definition

The factors which are important for simulating p2p environments are the overlay topology, the profile management of each individual peer, the nature of distribution of these profiles and the affinity measure based upon which the search algorithm is developed. Each of these factors is discussed one by one.

**Topology :** The overlay topology responsible for maintaining the neighborhood connections between the peers in the p2p network is considered to be a toroidal

grid where each node in the grid is conceived to be hosting a member (peer) of the p2p network. Due to the grid structure, each node has a fixed set of eight neighbors. A peer<sup>2</sup> residing in a particular node has correspondingly eight neighbors. Each peer carries two profiles - the *informational profile* and the *search profile*. The concept of *information* and *search* profile is explained next.

**Profile :** The *informational profile* ( $P_I$ ) of the peer is formed from the information which it shares with the other peers in the p2p network. The *search profile* ( $P_S$ ) of a peer is built from the informational interest of the user; formally it is represented in the same way as is  $P_I$ . In general, the search profile may differ from the information stored on the peer. For instance, a peer may contain some information about scientific researches in the field of distributed networks, but, in addition, the user may also be interested in football. For simplicity we assume that there are 1024 coarse-grained profiles, and let each of these profiles be represented by a unique ( $d =$  ) 10-bit binary token. The query message packet ( $M$ ) is also a 10-bit binary token. From now on we interchangeably use the term profile and token. Similarity between a profile  $P$  and a query message packet ( $M$ ) is measured by the number of bits that are identical. That is,  $sim(P, M) = d - HD(P, M)$ , where  $HD$  is the *Hamming distance* between  $P$  and  $M$ . Zipf's distribution[7], is chosen to distribute each of the 1024 unique alternatives in the network. The ranking of tokens in terms of frequency is the same for both information and search profiles.

We now present the search algorithm *ImmuneSearch*.

## 2.2 ImmuneSearch

The *ImmuneSearch* is a distributed algorithm and will be executed individually and independently by each peer. The search algorithm consists of two parts, the dynamics of packet movement through the network and the topology evolution initiated as a result of search.

**Packet Movement :** The search in our p2p network is initiated from the user peer. The user ( $U$  in *Fig. 1*) emanates message packets ( $M$ ) to its neighbors - the packets are thereby forwarded to the surroundings. The message packets ( $M$ ) are formed from the search profile  $P_S$  of  $U$ . The method of spreading message packets forms the basis of the algorithm.

In the system, the packets undergo a random walk on the grid, but when they come across a matching profile (information profile of any arbitrary peer), that is, the similarity between a message packet and informational profile is above a threshold, the message packet undergoes proliferation (as around peer  $A$  of *Fig. 1*), so as to find more peers with similar information profile around

---

<sup>2</sup> Although, in standard literature, 'peer' and 'node' are synonymous terms, the terms have been differentiated in the paper for ease of understanding. Node here means a position in the grid and essentially indicates a neighborhood configuration. A peer entering the network is assigned a node by the overlay management protocol. During topology evolution (discussed next) peers occupy new nodes and acquire new sets of neighbors.

the neighborhood. Some of the proliferated packets are also mutated. (Cf. the message packets distinguished by different gray levels around  $A$  in *Fig. 1*). Due to mutation the chance of message packets meeting similar items increases, which in turn helps in packet proliferation.

In order to realize the above described processes (random walk, proliferation and mutation) of message packets, each peer, executing the algorithm locally, should react when a message packet ( $M$ ) enters the node. The algorithm named *Reaction\_p2p* is executed by a peer  $A$ , with information profile  $P_I$ , when it encounters a packet ( $M$ ).

**Algorithm 1** *Reaction\_p2p*( $A$ )

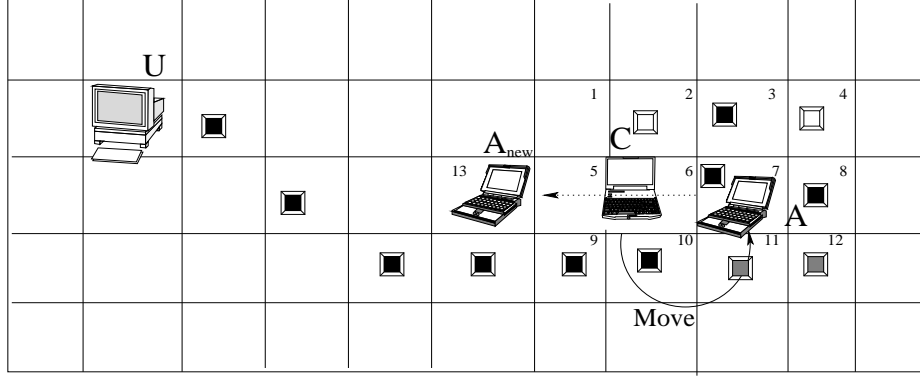
*Input* : Message packet( $M$ )

If ( $\text{sim}(P_I, M) \geq \text{Threshold}(\text{Pro}/\text{Mut})$ ) /\* $\text{Threshold}(\text{Pro}/\text{Mut})$  - Similarity threshold required to launch Proliferation/Mutation \*/  
    {Proliferate the packet  $M$  in the neighboring peers  
    Mutate some of the proliferated packets  $M$  to  $M'$ }  
else {Send the packet  $M$  to a randomly chosen neighbor peer}

It is clear from the algorithm, that the effect of proliferation and mutation is to initiate an intensified search around the neighbors of the peers which are already found to be similar to the queried profile. This implicitly points to the importance of topology evolution of the network, which should ensure that peers which have similar profiles come close to each other. This clustering would naturally increase the effectivity of the search, because packets after proliferation will immediately begin to find peers with similar information profiles, thus enhancing the efficiency of search.

**Topology Evolution :** In the topology evolution scheme, the individual peers change their neighborhood configuration during search so as to place them ‘closer’ to  $U$ . *Fig. 1* illustrates the exact mechanism of this movement. In the figure, peer  $A$  moves (changes its neighborhood configuration) from node 7 to node 13 to place itself ‘closer’ to  $U$ . Correspondingly, other peers adjust their positions. We now explain the factors based upon which a peer decides to change position as well as the rules guiding the degree of change.

A peer (say  $A$ ) decides to change its neighborhood configuration and places itself ‘closer’ to the user, when similarity between the profile of peer ( $P$ ) and the message ( $M$ ) sent by user peer (say  $U$ ) is above a threshold level. The amount of movement of  $A$  towards the user peer ( $U$ ) is determined by several factors. The distance moved is proportional to (a) the similarity between them ( $P$  and  $M$ ) either in terms of (i)  $P_I$  or (ii)  $P_S$ ; and (b) proportional to the distance between node  $U$  and node  $A$ . (c) The movement is also controlled by a further important process which is inspired by natural immune systems - *aging*. The movement of a peer gets restricted as it ages. The age of a peer is determined in terms of the number of times it undergoes movement as a result of encountering similar message packets. That is, the longer it stays in the environment (p2p network), the more it is assumed that the peer has found its correct node position, and hence the less it responds to any call for change in neighbors towards any user peer  $U$ . If the search profile ( $P_S$ ) of peer  $A$  matches



**Fig. 1.** Search Mechanism (packets passing and topology evolution). The figure shows topology evolution of a particular peer  $A$ , which moves to new position  $A_{new}$  after evolution. To do so, it swaps its position with intermediate nodes (for example  $C$ ). Other peers undergo identical evolution mechanism when query packets reach them.

$M$ , but peer ( $A$ ) has performed the search operation more times than  $U$ , then there is no movement of peer  $A$  towards user peer  $U$ . The aging concept lends stability to the system; thus a peer entering the p2p network, after undergoing initial changes in neighborhood, finds its correct position.

**Algorithm 2** *Topology\_Evolution(A)*

*Input : Message packet(M)*

*If ( $[sim(P_I, M) \text{ or } sim(P_S, M)] \geq d$ ) /\*  $d$  - Similarity threshold required to perform topology evolution \*/*

```

{  $x_1 = sim(P_I, M)$  /*  $P_I$  - information profile of A */
 $x_2 = age(A)$ ;  $x_3 = dist(A, U)$  /*distance between A and U*/
 $x_4 = sim(P_S, M)$  /*  $P_S$  - search profile of A */
 $x_5 = search\_performed(A) - search\_performed(U)$  /*search_performed(X) -
No of times X has previously performed search*/
if ( $x_1 > Threshold$ )
 $mov(A) \propto \frac{x_1 \cdot x_3}{x_2}$  /*mov(A) - Amount of movement of A towards U*/
else if ( $x_4 > Threshold$  and  $x_5 < 0$ )
 $mov(A) \propto \frac{x_4 \cdot x_3}{x_2}$  }
```

We are now in a position to present the main algorithm *ImmuneSearch* which is a combination of *Algorithms 1 & 2*. Each peer applies the *ImmuneSearch* algorithm whenever it encounters a message packet.

**Algorithm 3** *ImmuneSearch(A)*

*Input : Message packets (M)*

*Output : Search Result*

*if ( $sim(P_I, M) \geq d - 1$ )/\* Similarity threshold for a successful search \*/*

*Output ("Successful Search")*

*Topology\_Evolution(A)*

*Reaction\_p2p(A)*

### 3 Simulation Results

The experimental results, besides illustrating the efficiency of the *ImmuneSearch* algorithm, also show the self-organizing capacity of the algorithm in face of heavy unreliability of the peers participating in a p2p network. For comparison, we also simulate experiments with a random walk, two schemes of proliferation/mutation termed *proliferation<sub>1</sub>* and *proliferation<sub>2</sub>*, as well as a simple flooding technique. In *proliferation<sub>1</sub>* and *proliferation<sub>2</sub>*, peers basically execute the *ImmuneSearch* algorithm without the *Topology-Evolution* step. The threshold conditions applied to the two schemes differ; this point will be discussed later.

#### 3.1 Experimental Setup

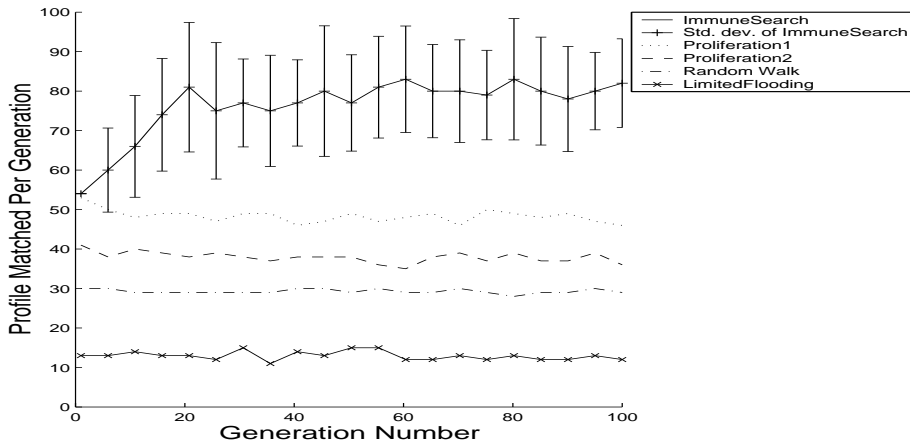
To understand the effect of proliferation and mutation rates, experiments with different rates and different threshold values (Threshold(Pro/Mut) - Algorithm 1) have been performed. From these, we report two cases which represent two main trends observed by us. In both cases, the proliferation and mutation rate is the same; however, the value of Threshold(Pro/Mut) differs. For the first case, Threshold(Pro/Mut) is  $(d - 1)$ ; while in the second case, it is  $(d - 2)$  ( $d$  is the length ( $= 10$ ) of the token). *ImmuneSearch* and *proliferation<sub>1</sub>* represent the first case, while *proliferation<sub>2</sub>* represents the second case. The number of packets proliferated ( $NR$ ) in the neighborhood is given by the following equation -  $NR = 8 \cdot S$ , where  $S = \frac{sim(P_I, M)}{d}$ ; while the probability of each of those  $NR$  packets undergoing one bit mutation ( $MP$ ) is 0.05.

Each search is initiated by a peer residing at a randomly chosen node and the number of search items ( $n_s$ ) found within 50 time steps from the commencement of the search is calculated. A generation is defined as a sequence of 100 searches. The search output ( $n_s$ ) is averaged over 100 different searches (a generation), whereby we obtain  $N_s$ .

In the graphs (*Fig. 2 & 4*) we plot this average value  $N_s$  against generation number to illustrate the efficiency of different models. We perform two types of experiments within the above mentioned experimental setup. In the first experiment, no peers leave the system, while the second experiment represents a more transient situation where peers leave/join the network at random.

#### 3.2 Expt. I : Search in Stable Conditions

This experiment is carried out under the assumption that no peer leaves the system. The system is represented by a  $100 \times 100$  toroidal grid, that is, it consists of  $10^4$  peers. We have initiated experiments with random walk, two types of proliferation/mutation schemes (*proliferation<sub>1</sub>* and *proliferation<sub>2</sub>*), limited flooding, and *ImmuneSearch*. The graph of *Fig. 2* displays the performance of the five different models. The  $x$ -axis of the graph shows the generation number while the  $y$ -axis represents the average number of search items ( $N_s$ ) found in the last 100 searches. The performance comparison of the above mentioned five methods obeys fairness criteria with respect to 'power' which are discussed next.

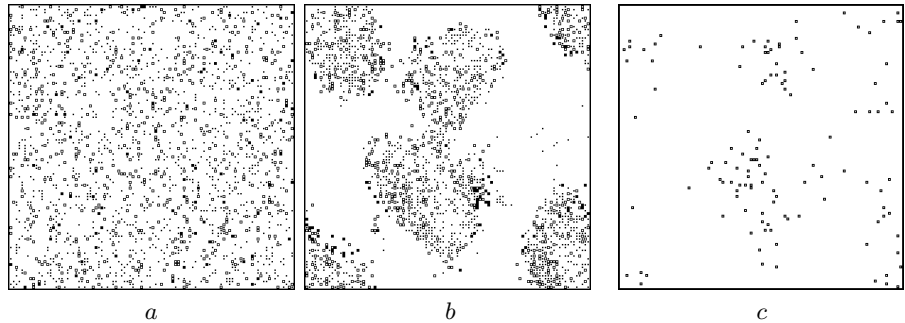


**Fig. 2.** Efficiency of different techniques of search namely *ImmuneSearch*, *proliferation<sub>1</sub>*, *proliferation<sub>2</sub>*, random walk and limited flooding. (Search results are averaged over 20 simulation runs). Standard deviation of the results is also shown for *ImmuneSearch* algorithm.

To provide fairness in ‘power’, two different approaches are taken. The first approach defines fairness among *ImmuneSearch*, *proliferation<sub>2</sub>*, random walk, and limited flooding, while the second approach defines fairness between *ImmuneSearch* and *proliferation<sub>1</sub>*. The initial conditions (number of message packets) for *ImmuneSearch*, *proliferation<sub>2</sub>*, and random walk, are chosen in a way such that the total number of packets used over 50 time steps of each individual search is roughly the same. In the case of flooding, we have allowed the process to run for  $x$  number of steps where  $x (< 50)$  steps uses the same number of packets as the aforesaid three cases used in 50 time steps. The fairness in ‘power’ between *Proliferation<sub>1</sub>* and *ImmuneSearch* is maintained by employing the same threshold level for proliferation, and the same proliferation/mutation rate.

**Search Efficiency :** In *Fig. 2*, it is seen that the number of search items ( $N_s$ ) found is progressively higher in limited flooding, random walk, *proliferation<sub>2</sub>*, *proliferation<sub>1</sub>*, and *ImmuneSearch*, respectively. The *proliferation<sub>1</sub>*, *proliferation<sub>2</sub>*, random walk, and limited flooding maintain a steady average search output of around 50, 40, 30, and 15 hits respectively. The standard deviation of the output is roughly around 10% of mean in each case. In the *ImmuneSearch* algorithm, it is observed that after it starts at an initial output of around 55 items per search, it steadily increases to 80 within the 25<sup>th</sup> generation, and then maintains a steady output of about 80 per search.

The difference in performance among the first four schemes which don’t undergo any topology evolution can be directly attributed to the different frequencies, with which multiple message packets visit the same peer. The probability is particularly high in limited flooding where each message packet at every instance floods packets to all its neighbors. Therefore, at every instance, a multi-



Clustering of information (small dots) and search (big dots) profile of peers possessing most frequent tokens at generation no. 0, 24 respectively. Information profile of peers hosting 11<sup>th</sup> most frequent tokens at generation 100.

**Fig. 3.** Snapshots showing clustering of similar peers in the p2p network.

ple number of packets visits the same peer. In random walk, the probability of multiple packets visiting the same peer is high at the beginning of the search when they are clustered around the neighborhood of the user peer. The proliferation/mutation scheme in this respect has an advantage, in that initially it starts with a lower number of packets; only when they are sufficiently far apart, they do proliferate. Thus the probability of multiple packets visiting the same region is low. Proliferation/mutation schemes can also control the number of packets produced according to requirement. Since proliferation is dependent upon affinity between the message packets and the information profile, typically a lower number of packets is produced when the searched item is sparse in the system. This stops unnecessary wastage and in turn improves the effectivity of message packets. However, the experiments show that merely more proliferation does not necessarily imply a high rate of success, but that regulation of the proliferation/mutation scheme is also very important. It is seen that although *proliferation<sub>2</sub>* produces more message packets than *proliferation<sub>1</sub>*, the search success rate is higher in *proliferation<sub>1</sub>* than *proliferation<sub>2</sub>*.

In *ImmuneSearch*, the first 25 generations can be termed as ‘learning’ phase. During this time, similar to natural immune systems, the p2p network *develops memory* by repositioning the peers. The repositioning results in clustering of peers with similar profiles which is discussed next.

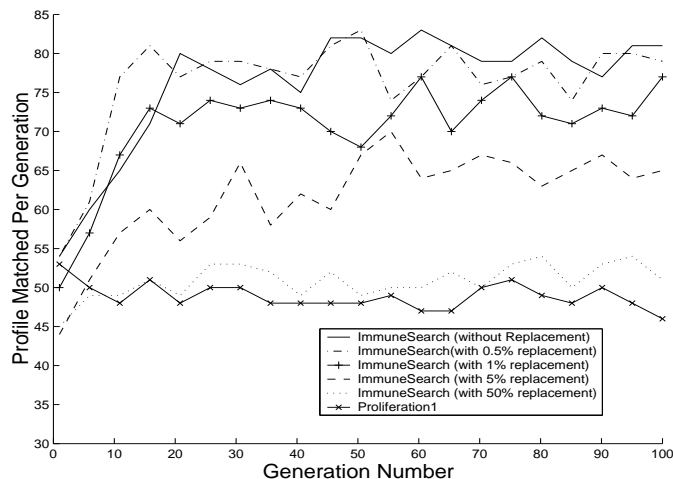
**Clustering Impact :** The series of snapshots in *Fig. 3* demonstrates the clustering effect in the p2p network as a result of *ImmuneSearch*. Each figure represents the configuration on the  $100 \times 100$  overlay grid taken to host the 10,000 peers. In *Fig. 3a. & b.*, each peer displays its two profiles  $P_I$  and  $P_S$ . (The big dots represent the search profile of a peer ( $P_S$ ) while the small dots are the informational profile ( $P_I$ )). In *Fig. 3c.*, we show only the informational profile represented as dots.

The second snapshot (*Fig. 3b*) exhibits the clustering of the most frequently occurring profile at generation number 24 (the generation around which ‘learn-

ing’ is more or less complete) from the initial scattered setting (*Fig. 3a*). The snapshot of generation 24 shows that peers with search profile  $P_S$  intermingle with the peers with information profile  $P_I$ . That is, execution of the algorithm results in the peers with search profile ( $P_S$ ) positioning themselves in ‘favorable’ positions whereby, when these peers initiate a search, the message packets emanated by them immediately begin to find peers with similar profiles. The clustering of the peers, as seen in *Fig. 3b*, is roughly divided into four major clusters; and it is notable that the clusters are porous. The porous and separated clusters are a result of ongoing competition among the differently frequent tokens and as a result of it also less frequent tokens obtain space to form clusters. Subsequently, their search output is also enhanced. (As an example, *Fig. 3c* shows clusters of peers hosting the 11<sup>th</sup> most frequent token).

### 3.3 Expt II : Search in Transient Conditions

The robustness of the algorithm is demonstrated by the following experiment. In this experiment, 0.5%, 1%, 5%, or 50% of the population, respectively, is replenished after every generation. This mimics the transient nature of p2p networks where peers regularly join and leave the system. *Fig. 4* shows the performance of the *ImmuneSearch*, under various degrees of replacement.



**Fig. 4.** Performance efficiency of ImmuneSearch when 0%, 0.5%, 1%, 5%, and 50% of peers are respectively replaced after each generation. The performance efficiency of *proliferation<sub>1</sub>* is also plotted as a base case. (Results are avg. of 20 simulation runs).

The results of *Fig. 4* illustrate two important aspects. (i) First of all, even in face of dynamic change, the *ImmuneSearch* algorithm ‘learns’, in that, after some initial generations, the efficiency increases. The rate of increase in search

efficiency during the initial generations is generally dependent on the amount of replacement the p2p network undergoes after each generation. We find that the performance of *proliferation*<sub>1</sub> is roughly the same when there is 50% replacement. But replacement of 50% of all peers in only 100 searches is likely far higher than any realistic turnover rate. (ii) However, the more important point to be noted is that at 0.5% replacement, we observe that the performance is in fact at par and sometimes slightly better than *ImmuneSearch* without replacement! The result establishes one important advantage of the algorithm - that is, a little transience is helpful, rather than detrimental, to the performance of the algorithm. This happens because the problem of developing a search algorithm is in fact a multi-objective optimization problem, and due to the enormous complexity, we are obtaining a 'good', however not optimal solution. So a little change in peers is probably enabling the system to move quickly towards a better solution.

## 4 Conclusion

This paper has presented a search algorithm which derives its inspiration from natural immune systems. The beauty of the algorithm lies in its simplicity. However, this simple decentralized algorithm generates emergent properties like a *complex adaptive system*, whose underlying guiding rules are generally also very simple. We find that as a result of the algorithm, the p2p network 'learns' and subsequently develops memory, whereby the search efficiency improves dramatically after some initial learning/training phase. The basic strengths displayed by the *ImmuneSearch* algorithm need to be further explored and developed, by applying it in more realistic circumstances in the near future.

## References

1. G Canright, A Deutsch, M Jelasity, and F Ducatelle. Structures and functions of dynamic networks. Bison Deliverable, [www.cs.unibo.it/bison/deliverables/D01.pdf](http://www.cs.unibo.it/bison/deliverables/D01.pdf), 2003.
2. D. Dasgupta and S. Forrest. Novelty Detection in Time Series Data using Ideas from Immunology. In *ISCA 5<sup>th</sup> International Conference on Intelligent Systems*, 1996.
3. J. E. Hunt and D. E. Cooke. Learning Using an Artificial Immune System. *Journal of Network and Computer Applications*, 19:189–212, 1996.
4. J. O. Kephart. A Biologically Inspired Immune System for Computers. In *Proceeding of Artificial Life*, July 1994.
5. Q. Lv, P. Cao, E. Cohen, and S. Shenker. Search and Replication in Unstructured Peer-to-Peer Networks. In *Proceedings of the 16<sup>th</sup> ACM International Conference on Supercomputing*, June 2002.
6. S. Singh and S. Thayer. A Foundation for Kilorobotic Exploration. In *Proceedings of the Congress on Evolutionary Computation at the 2002 IEEE World Congress on Computational Intelligence*, May 2002.
7. G. K. Zipf. *Psycho-Biology of Languages*. Houghton-Mifflin, 1935.