Scalability Analysis: Reconfiguration of Overlay Networks Using Nature-Inspired Algorithms

Simone A. Ludwig

Abstract Overlay networks are virtual networks of nodes and logical links built on top of the existing network infrastructure, with the purpose of contributing new functionality. There are many different solutions proposed to tackle a range of specific needs such as content distribution and caching, file sharing, improved routing, multicast and streaming, ordered message delivery, and enhanced security and privacy. In this chapter, the focus lies on the optimization of overlay networks in terms of cost, performance, and reliability. In particular, the main objective is the optimization of data mirroring. Three different optimization approaches are introduced. The first approach is based on a ”related work” implementation using Genetic algorithms, the second makes use of artificial immune systems, and the third approach uses the Particle swarm optimization approach. The three algorithms are implemented and experiments are conducted to measure the overall performance, the behavior and feasibility of network and link failures, as well as a scalability analysis is performed.

1 Introduction

Many different kind of overlay networking technologies have emerged in the past years. Research and development of overlay systems have primarily focused on developing technologies that solve the challenges of reliability and efficient processing of networks by providing a higher-level network that is built on top of the normal network, the so-called overlay network. As the overlay network is built on top of an existing network, it relies on the underlay network for basic networking function such as routing and forwarding. The nodes in an overlay network are connected via logical links and can span many physical links.

Simone A. Ludwig
Department of Computer Science, North Dakota State University, Fargo, ND, USA, e-mail: simone.ludwig@ndsu.edu

In particular, the huge number of Internet users, estimated to almost 2 billions to date [1], as well as the delivery of huge amounts of data and media has become commonplace. Multimedia content such as videos is posing an increasing challenge to the networks, as well as social collaboration and social media web sites, which use and distribute large amounts of data on a daily basis. These developments of the evolving web have a profound impact on the network requirements in terms of performance and reliability. Therefore, it is essential that overlay networks have to ensure an efficient and scalable service for Internet users.

An overlay network built on top of an existing network consists of a set of distributed nodes that are deployed on the Internet. The nodes of the overlay network are expected to meet the following requirements [2]:

- Provide the infrastructure to support the execution of one or more distributed applications.
- Support high-level routing and forwarding tasks necessary in a network. The overlay network should provide data-forwarding capabilities that are different from the ones that are part of the Internet.
- Deployment should be across the Internet to allow third parties to participate in the organization and operation of overlay networks.

Overlay networks have many advantages that can be listed as such [3]:

- Overlay networks are incrementally deployable - The overlay network does not require changes to the existing Internet infrastructure, only additional servers are necessary. Once nodes are added to an overlay network, the control of paths of data becomes possible with great precision.
- Overlay networks are adaptable - Even though the abstraction of an overlay network constrains the packets to flow over a constrained set of links, the set of links can be constantly optimized over metrics that the different applications require.
- Overlay networks are robust - Robustness is a result of the given increased control and adaptable nature of the overlay networks. With a sufficient number of nodes deployed, an overlay network should be able to route between any two nodes in two independent ways, i.e., overlay networks are able to route around faults occurring in the network.
- Overlay networks are customizable - Given that overlay nodes can be multipurpose computers, they can be easily equipped with whatever is necessary. To give an example, overlay networks make extensive use of disk space that allows overlay networks to provide savings in terms of bandwidth when the content is not consumed simultaneously in different parts of the network.

The main properties of overlay networks are adaptability and robustness. These two features are the major diving force behind the research of overlay networks. The aim of this paper is the optimization of an overlay network in terms of cost, performance, and reliability. In particular, the main focus of the optimization is the application of data mirroring. One ”related work” approach, as well as two additional approaches are compared in terms of performance and scalability. Part of this research presented in this book chapter was published in [4].

The remainder of this chapter is as follows: Section 2 describes related work; Section 3 introduces the approaches used; in Section 4 the experimental setup and results are described; and in Section 5 the findings are summarized.

2 Related Work

There are many different types of overlays that were developed meeting a range of purposes, e.g. content distribution and caching [5], overlays used for file sharing [6], improved routing [7], multicast and streaming [8], ordered message delivery [9], and enhanced security and privacy [10].

In the next few paragraphs we list and describe in more detail particular overlay network implementations that are developed. A routing overlay network is an overlay that controls and/or modifies the path of data delivery through the network. Routing overlay networks [7] improve the performance and robustness of packet delivery. This improvement is achieved by delegating the task of selecting paths to users, who can choose among more reliable routes, less loaded routes, shorter routers, or higher bandwidth routes. Overlay networks do not require support from routers as other path selection methods do. The packets still traverse the underlying routing infrastructure, however, their path is defined by the logical edges of the overlay.

Another approach is called Overcast [11], which is an application-level multicast system that can be incrementally deployed making use of the Internet infrastructure. Basically, the implementation consists of a collection of nodes that are placed at strategic locations in an existing network, which in turn implement a network abstraction on top of the network provided by the underlying network. Overcast provides multicast that is scalable and reliable by using a simple protocol for building efficient data distribution trees that automatically adapt to changing network conditions. The simulations conducted indicate that the Overcast provides roughly 70-100% of the total bandwidth possible, at a cost of somewhat less than twice the network load.

Another well-known overlay network implementation is RON (Resilient Overlay Network) [12]. RON is an architecture that allows distributed Internet applications to detect path errors/outages and recover from them within seconds, thus improving wide-area routing protocols that take several minutes to recover. It is an application-layer overlay on top of the existing Internet routing substrate. RON monitors the quality of the Internet paths in order to decide whether a route change of the packets need to take place to improve the overall quality of the overlay network. RON was able to improve the loss rate, latency, or throughput perceived by data transfers (5% of the transfers doubled their TCP throughput, and 5% of the transfers had the data loss reduced by a factor of 0.05).

Research concerned in particular with the optimization of overlay networks is manifold. One special class of overlay networks is the Service Overlay Networks (SON) [13]. The aim is to design the overlay with reliability constraints in mind.

The assumption is that a SON could enter an inadmissible state for two reasons; (1) due to insufficient resources to accommodate new connections, and (2) due to some hardware malfunctioning. Two sub-functions, which are Maximum Profit (MP) and Minimum Cost (MC) need to be optimized. Given the competing relationship between MP and MC, the aim is to ensure the systems' operability. Their approach is based on the Lagrange multipliers, which shows that MP and MC separately can achieve the same network designs. The paper's contribution is the provisioning of guidelines to the formulation of the design of a reliable and economically optimal SON.

Given that the optimization of an overlay network configuration is NP-hard, classic exact techniques are only applicable to solve very small instances of overlay networks. Among a broad set of modern heuristics and metaheuristics methods for optimization, nature-inspired methods have emerged as promising techniques for solving for example network design problems since they are able to compute approximate solutions within acceptable range of execution times [14].

The research and developments of nature-inspired networking techniques have fostered new techniques in networking, in particular due to their dynamic nature, resource constraints and heterogeneity. In particular, an Ant Colony Optimization approach was used in the AntNet routing protocol [15]. Their protocol uses agents to concurrently explore the network and exchange collected information in the same way as ants explore the environment. The main idea used from the Ants is their indirect communication capability depositing pheromone for the routing optimization.

Multi-objective evolutionary algorithms, in particular, NSGA-II was used in [16] to optimize a multicast overlay network based on two criteria; the first is to optimize the total end to end delay of a multicast tree, and the second is to maximize the link utilization.

A swarm-intelligence based approach was used in a layered overlay multicast approach for routing web streams [17]. An architecture was adopted to improve service capabilities, for satisfying the request of multi-constrained Quality of Service (QoS) routing of large-scale multi-domain web streams. The approach is based on meeting the uncertainty of the network status description, in order to find the QoS-satisfied routes using an effective mathematical model.

A multi-swarm approach for neighbor selection in peer-to-peer overlay networks is described in [18]. Their approach is inspired by the commonalities of P2P systems and Particle Swarm in a dynamic environment. A multi-swarm interactive pattern was introduced to match the dynamic nature of P2P networks.

In [19], genetic algorithms are applied to the reconfiguration of the topology and link capacities of an operational network. It does this in order to adapt to changes of its operational conditions, in which nodes and links might become unavailable, the traffic patterns might change, and the quality of service requirement and priorities of different users and applications might change suddenly.

Another example of a genetic algorithm approach is the Genetic-Algorithm-Based Neighbor-Selection Strategy for Hybrid Peer-to-Peer Networks [20]. The strategy enhances the decision process applied to transfer coordination. An investigation of the strategy revealed that it affects the system throughput and distribution efficiency, as well as peer contribution, especially for low-connectivity peers.

Plato is a genetic algorithm approach to run-time reconfiguration in autonomic computing systems [21]. The genetic algorithm approach uses the evolutionary computation technique to automate the decision-making process of an autonomic system. It enables a system to dynamically evolve target reconfigurations at run time, and at the same time, it balances the trade-offs between functional and non-functional requirements to changes in the environmental requirements and conditions. In particular, their approach is applied to the reconfiguration of a collection of remote data mirrors demonstrating it to be a good optimization method for diffusing data, and minimizing operational costs; at the same time it maximizes data reliability and network performance.

This paper closely follows the Plato approach using genetic algorithms to optimize an overlay network based on cost, performance, and reliability. However, it varies in the following two aspects (1) the fitness functions are normalized in order to compare the three different measures, and (2) besides the use of genetic algorithms, an algorithm based on Artificial Immune Systems, and another based on Particle Swarm Optimization (PSO) are implemented and evaluated. As can be seen in the evaluation section, the two additionally implemented algorithms perform well, and in particular the PSO approach shows a better performance than the Genetic algorithm approach. Furthermore, we also address the issue of scalability of the three nature-inspired algorithms, and comment on the feasibility of the approaches for the automatic reconfiguration of overlay networks.

3 Approaches

This research addresses the dynamic reconfiguration of a collection of remote data mirrors. In remote data mirroring, data copies of critical data is stored at one or more secondary site(s), which prevents the protected data from failures that may affect the primary copy [22]. There are two important design criteria for remote data mirroring; the first is to choose the type of network link connection to the mirrors, and the second is to choose the remote mirroring protocol. Each link in the network has a cost associated, as well as throughput, latency and loss rate which determine the overall remote mirror design performance [22]. There are two types of remote mirroring protocols that are synchronous and asynchronous propagation, both affecting network performance and data reliability. In synchronous propagation the secondary site applies each write before the write completes at the primary site, and in asynchronous propagation updates get queued at the primary site and are periodically propagated to the secondary site in a batch-fashion.

The optimization design criteria are the same as for Plato [21]. The main goal is the construction and maintenance of an overlay network for the data to be distributed to all nodes while fulfilling the following requirements:

1. Overlay network must remain connected at all times;

2. Overlay network should never succeed the allocated monetary budget;
3. The data should be distributed as efficiently as possible, meaning the amount of bandwidth consumed when diffusing data should be minimized.

Three algorithms have been implemented in order to evaluate and compare their performance. The first algorithm is an implementation of Genetic Algorithms as in the Plato implementation [21], the second one is an implementation of the aiNet algorithm inspired by Artificial immune systems, and the third algorithm is an implementation of the discrete Particle Swarm Optimization approach.

Table 1 Link Propagation Methods

Time interval	Avg. data batch size in GB
0	0
1 min	0.0436
5 min	0.2036
1 hr	2.091
4 hr	6.595
12 hr	15.12
24 hr	27.388

The fitness function used as a measure for all three algorithms are slightly modified compared to the Plato approach [21]. The differences are that normalization of the overall fitness value is done in order to have an overall fitness value in the range of 0 and 1, as well as the sum of weights for each part of the fitness function is 1.

The fitness function consists of three parts (as in Plato); the first part evaluates the overlay network in terms of cost, the second in terms of performance, and the third part evaluates the reliability of the overlay network. The overall fitness function (Eq. (1)) is the weighted average of all three fitness portions. Please note that the sum of the weights needs to sum up to 1 (Eq. (2)).

$$F_{overall} = w_1 * F_{cost} + w_2 * F_{perf} + w_3 * F_{rel} \tag{1}$$

$$\sum_{i=1}^{3} w_i = 1 \tag{2}$$

Looking at the different fitness sub-functions, the fitness sub-function for cost is given as:

$$F_{cost} = 1 - \frac{cost}{budget} \tag{3}$$

where *cost* is the sum of operational expenses of all active links and *budget* is a user supplied value on the maximum amount of money for an operating overlay network. The sub-function for the performance consists of two parts: latency and bandwidth as given below:

$$F_{perf} = 0.5 * (1 - \frac{latency_{avg}}{latency_{wc}}) + 0.5 * (\frac{bandwidth_{sys} - bandwidth_{eff}}{bandwidth_{sys}} + bound) \quad (4)$$

where $latency_{avg}$ is the average latency over all active links, and $latency_{wc}$ is the largest latency value measure over all links in the underlying network; $bandwidth_{sys}$ is the total available bandwidth across the overlay network given the active links, and $bandwidth_{eff}$ is the total effective bandwidth across the overall network after data has been coalesced, and *bound* is a limit on the best value that can be achieved throughout the network.

The last fitness sub-function measures the overlay network in terms of reliability consisting of two parts as given below:

$$F_{rel} = 0.5 * (\frac{links_{used}}{links_{max}}) + 0.5 * (1 - \frac{dataloss_{pot}}{dataloss_{max}}) \quad (5)$$

where $links_{used}$ is the number of active links, and $links_{max}$ is the maximum number of possible links given the network structure; and $dataloss_{pot}$ is the total amount of data that could be lost during write coalescing using the propagation methods as given in Table 1, and $dataloss_{wc}$ is the amount of data that could be lost during write coalescing using the propagation method with the largest time window.

3.1 Genetic Algorithm Implementation

Genetic algorithms [23] are a class of stochastic search algorithms based on biological evolution. In particular, the principles of the evolution via natural selection are applied, employing a population of individuals that undergo selection, as well as variation-inducing operators such as mutation and crossover. A fitness function is used to evaluate individuals.

The genetic algorithm in short works as follows: a population is created with a group of individuals that are created randomly. The individuals in the population are then evaluated. The evaluation function, called fitness function, gives the individuals a score based on how well they perform at the given iteration. Two individuals are then selected based on their fitness, i.e., the higher the fitness, the higher the chance of being selected. These individuals then "reproduce" to create one or more offspring(s), and afterwards the offsprings are mutated randomly. This continues until a suitable solution has been found or a certain number of iterations have passed.

Table 2 lists the parameters used for the implementation, which is identical to the one proposed in Ramirez et al. [21] with the exception of the modified fitness function as explained above. Two-point crossover is employed, as well as Tournament selection is used.

Table 2 GA parameters

Parameter	Value
Population size	100
Crossover	Two-point
Crossover probability	0.1
Mutation probability	0.05
Selection method	Tournament (k=2)

3.2 Artificial Immune System Implementation

An Artificial Immune System (AIS) models the natural system's ability to detect foreign cells in the body. It is a new computational paradigm with the ability to perform pattern recognition that is mainly applied to anomaly detection.

There are different views on how natural immune systems have been developed. These models include the classical view of lymphocytes that are used to distinguish between self- and non-self, clonal selection theory, danger theory, etc. The implementation we have adopted and applied is based on the clonal selection theory. According to Burnet's clonal selection theory [24], the immune system undergoes a selection mechanism during the lifetime of the individual. The theory states that activation of lymphocytes occurs when the binding with a suitable antigen happens. Once activated, clones of the lymphocyte are produced expressing identical receptors to the original lymphocyte that encountered the antigen. This process ensures that only lymphocytes specific to an activating antigen are produced in large numbers.

Based on this biological background, the clonal algorithm was introduced by de Castro and Timmis [25], named *aiNet*. In order to present the pseudo code of the *aiNet* algorithm the following terms needs to be introduced:

- *Network cell*: individual of the population; each cell is a real-valued vector in an Euclidean shape-space (in our case it is an integer-valued vector);
- *Fitness*: fitness of a cell in relation to an objective function to be optimized (in our case the optimization is a maximization problem);
- *Clone*: offspring cells that are identical copies of their parent cell. The offspring will further suffer a somatic mutation so that they become variations of their parent.

The pseudo code of the slightly adapted (the concept of affinity was not used as not applicable) optimization version of the *aiNet* algorithm is summarized as follows:

The parameters used in the AIS implementation are stated in Table 3.

Algorithm 1 AIS pseudocode

```
Randomly initialize a population of cells (the initial number of cells is not relevant)
while stopping criterion is not met do
    Determine the fitness of each network cell.
    Generate a number of clones for each network cell.
    Mutate each clone proportionally to the fitness of its parent cell, but keep the parent cell.
    Determine the fitness of all individuals of the population.
    for each clone do
        Select the cell with highest fitness and calculate the average fitness of the selected popula-
        tion.
    end for
end while
```

Table 3 AIS parameters

Parameter	Value
Number of cells	2
Number of clones	6

3.3 Particle Swarm Optimization Implementation

Particle Swarm Optimization (PSO) as introduced in [26], is a swarm based global optimization algorithm. The algorithm models the behavior of bird swarms searching for an optimal food source. The movement of a single particle is influenced by its last movement, its knowledge, and the swarm's knowledge. PSOs basic equations are:

$$x_i(t+1) = x_i(t) + v_{ij}(t+1) \tag{6}$$

$$\begin{aligned} v_{ij}(t+1) = w(t)v_{ij}(t) + c_1 r_{1j}(t)(xBest_{ij}(t) - x_{ij}(t))) \\ + c_2 r_{2j}(t)(xGBest_j(t) - x_{ij}(t))) \end{aligned} \tag{7}$$

where x represents a particle, i denotes the particle's number, j the dimension, t a point in time, and v is the particle's velocity. *xBest* is the best location the particle ever visited (the particle's knowledge), and *xGBest* is the best location any particle in the swarm ever visited (the swarm's knowledge). w is the inertia weight and used to weigh the last velocity, c_1 is a variable to weigh the particle's knowledge, and c_2 is a variable to weigh the swarm's knowledge. r_1 and r_2 are uniformly distributed random numbers between zero and one.

PSO is usually used on continuous and not discrete problems. In order to solve the discrete overlay network assignment using the PSO approach, several operations and entities have to be defined. The implementation follows in part the implementation for solving the traveling salesman problem as described in [27]. First, a swarm of particles is required. A single particle represents one overlay network, i.e., every particle's position in the search space must correspond to a possible overlay network. Velocities are implemented as lists of changes that can be applied to a particle (its vector) and will move the particle to a new position (a new overlay network). Changes are exchanges of values of the overlay network. Further, minus between two matches (particles), multiplication of a velocity with a real number, and the addition of velocities have to be defined. Minus is implemented as a function of particles. This function returns the velocity containing all changes that have to be applied to move from one particle to another in the search space. Multiplication randomly deletes single changes from the velocity vector, if the multiplied real number is smaller than one. If the real number is one, no changes are applied. For a real number larger than one, random changes are added to the velocity vector.

The PSO implemented uses guaranteed convergence, which means that the best particle is guaranteed to search within a certain radius, implying that the global best particle will not get trapped in local optima. Table 4 shows the specific parameters chosen for the implementation.

Table 4 PSO parameters

Parameter	Value
Number of particles	100
Inertia weight	0.001
Weight of local knowledge	0.5
Weight of global knowledge	0.5
Radius	2
Neighborhood size	4

4 Experiments and Results

The three algorithms were tuned with the parameter values given in the previous section. Since the GA and PSO implementation have probabilities involved, all parameters of the algorithms were set, so that the number of iterations needed were kept constant, but at the same time the number of function evaluations is kept as equal as possible. Furthermore, all experiments were conducted 25 times in order to account for statistical variations.

The first set of experiments was performed measuring the accuracy of all approaches given different settings of the weights, comparing the different effects on the accuracy. The second set investigates single link failures, and the third set evaluates complete network failures. The last set of experiments explores the effect of increasing overlay network sizes, and the impact on the accuracy, as well as the performance in terms of actual execution times.

4.1 Overall Comparison of Approaches

Table 5 shows the results of the fitness scores of all approaches for the configuration of a 25-node network using different weight combinations, thereby favoring cost, performance or reliability. The values were taken at iteration 500. It can be seen that the highest fitness score can be achieved with F2 and F3, which only consider the cost and performance fitness sub-functions respectively (a smaller number of links achieves a higher fitness score). The worst fitness score is observed for F4, when optimizing the overlay network based on reliability given the trade-off between the number of links and the potential data loss. Overall, PSO outscores the other approaches for all weight settings.

Table 5 Fitness function and fitness values (500 iterations)

Fitness function	w_1	w_2	w_3	Best fitness GA	Best fitness AIS	Best fitness PSO
F1	$0.\overline{3}$	$0.\overline{3}$	$0.\overline{3}$	0.8233	0.8069	0.8333
F2	1	0	0	0.8910	0.9021	0.9129
F3	0	0	0	0.8922	0.8965	0.9087
F4	0	1	1	0.6037	0.6053	0.6086
F5	0.5	0	0	0.8978	0.8796	0.8999
F6	0	0.5	0.5	0.7394	0.7337	0.7500
F7	0.5	0.5	0.5	0.7301	0.7297	0.7499

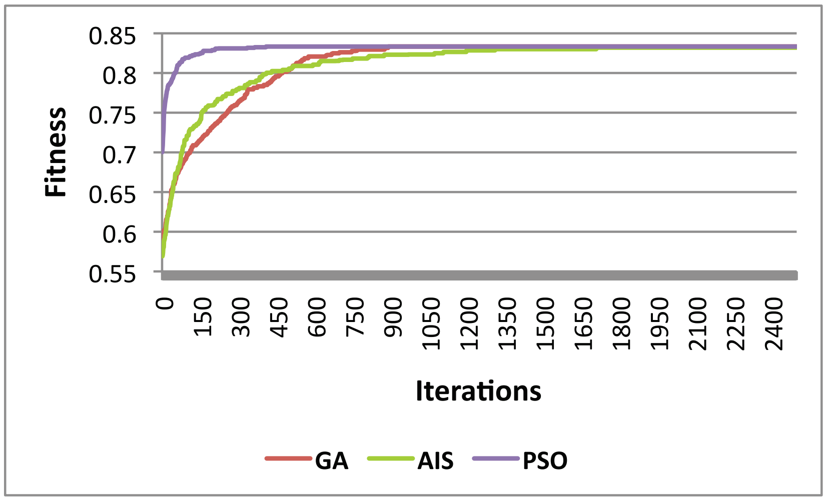

Fig. 1 Fitness of all approaches

Figure 1 shows the fitness values plotted for a 25-node network for increasing iterations. It can be observed that the PSO approach reaches the maximum fitness score of 0.8333 at iteration 270, which is much earlier than the GA and AIS approaches. The GA approach takes 610 iterations, and the AIS approach requires 1420 iterations to reach the maximum fitness score. The fitness function F1 with an equal weight distribution was used.

4.2 Investigation of Network and Link Failures

Figure 2 shows three complete reconfigurations of a 25-node overlay network. The simulation runs the optimization every 2,500 iterations, due to an artificially induced breakdown of the network. As can be seen in Figure 2, all three approaches, GA, AIS and PSO, can reconfigure overlay networks and achieve the maximum fitness score after around 1,200, 300 and 1,600 iterations respectively.

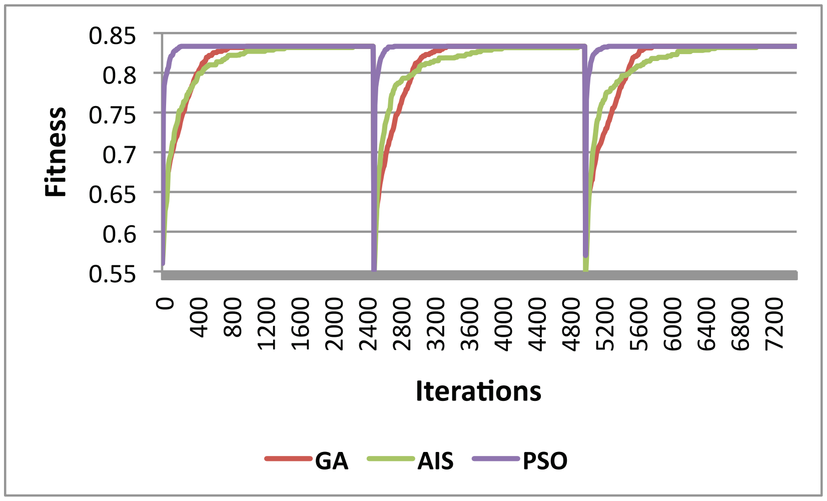

Fig. 2 Reconfiguration of complete overlay network with 25 nodes

Figure 3 shows the reconfiguration of a 25-node network. The first evolution shows the fitness trend when F1 is used (weights are all $0.\overline{3}$), after a failure occurs the network switches the weights to w_1=w_2=0.25 and w_3=0.5 in order to stress more on reliability rather than an equal contribution of all three fitness sub-functions. Even though the fitness score is lower with the second weight setting, however, the network configuration is more stable to protect against future failures.

Figure 4 shows the fitness curves for successive link failures without reconfiguration. It can be seen that the generation of new overlay network configurations work fine until 84% of link failures occur, however, at the cost of reducing fitness scores.

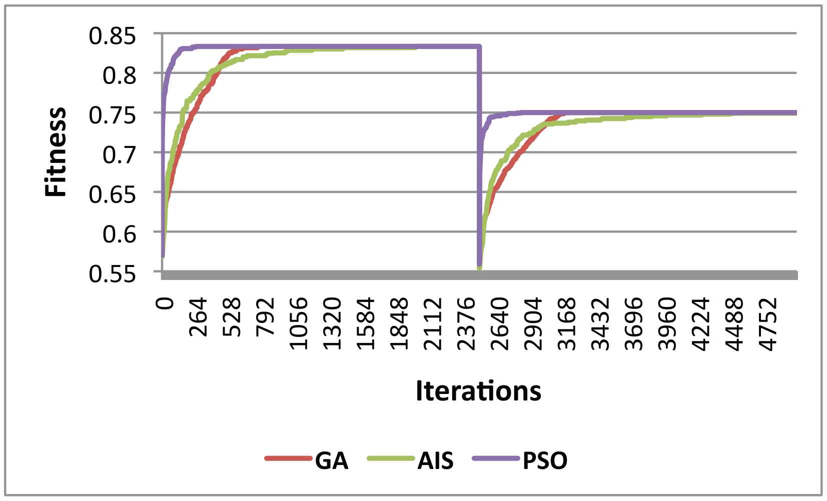

Fig. 3 Reconfiguration of 25-node network with varying weight distributions of fitness function

Decreasing fitness scores can be observed with the lowest of 0.34 when 80% of link failures occur; after 84% an overlay network cannot be constructed anymore, and therefore the reconfiguration needs to be started.

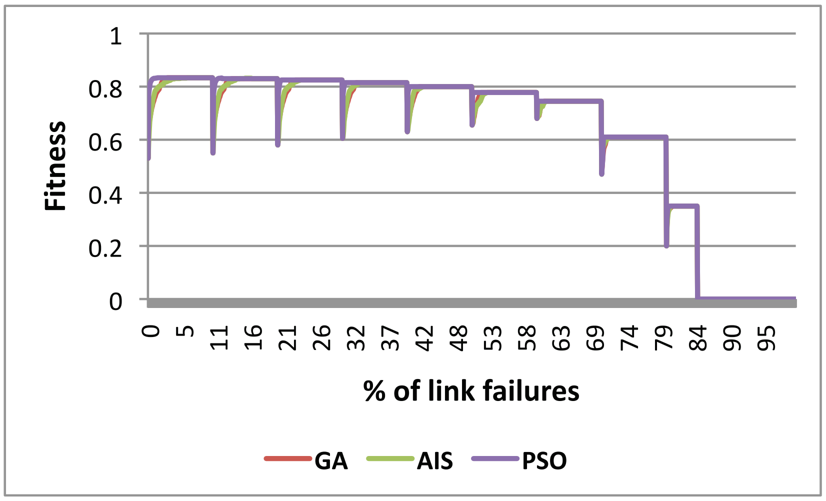

Fig. 4 Fitness scores with increasing percentage of link failures

4.3 Scalability Analysis

The scalability analysis investigates the performance impact with increasing network sizes. We successively increase the number of networks starting with 10 nodes all the way up to 60 nodes.

Figure 5 shows the fitness scores of the approaches for a network of size 60. What can be seen, compared to Figure 1, is that even though PSO achieves larger fitness improvements at the beginning, however, after around 2,700 iterations GA has already achieved the maximum fitness score, whereas it takes the PSO approach many more iterations to reach the maximum fitness score (89426 iterations on average). The AIS approach does not scale very well. Even though it achieves the maximum fitness score eventually; many more iterations are necessary compared to the PSO and GA approaches.

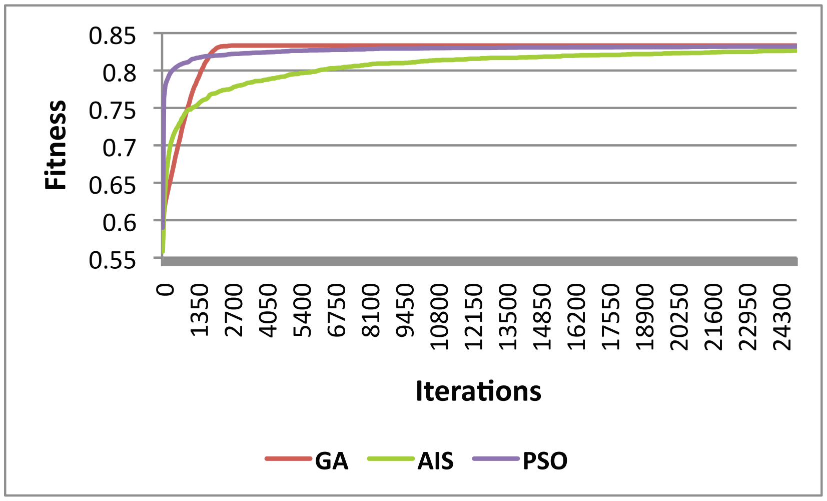

Fig. 5 Fitness of all approaches for network size of 60 nodes

Figure 6 shows the number of iterations it takes to achieve the optimal fitness scores for all approaches. It can be seen that the number of iterations necessary to achieve the optimum is first lower for PSO, but after a network size of around 40 nodes, GA reaches the maximum fitness values faster than PSO. For larger networks (above 50 nodes) AIS shows a slightly better fitness score than PSO. Please note that the y-axis is in logarithmic scale.

Figure 7 shows the same trend as Figure 6 in terms of execution time. However, the two lines for PSO and GA cross approximately at 43 nodes, whereas for AIS and PSO the cross happens after 50 nodes. Please also note that the y-axis is in logarithmic scale.

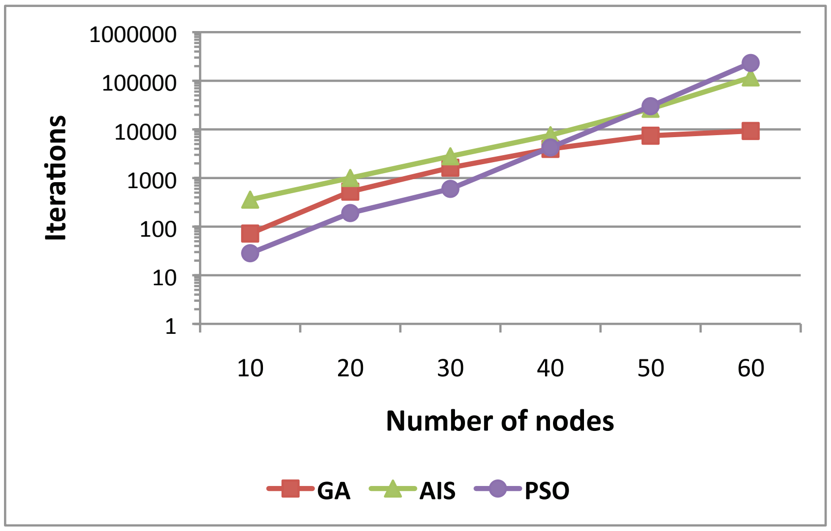

Fig. 6 Maximum fitness scores for different network sizes

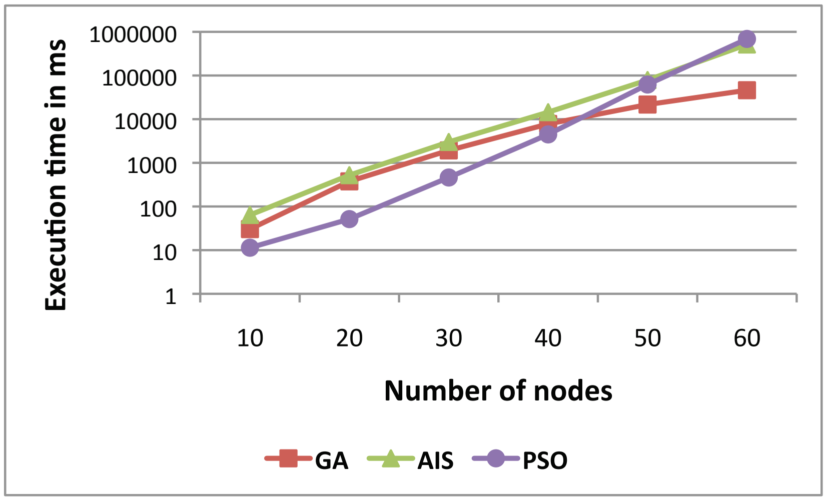

Fig. 7 Execution times in milliseconds of maximum fitness scores achieved (Figure 6)

5 Conclusion

This paper investigated three approaches for the reconfiguration of overlay networks and could be used as a guide for overlay network construction and configuration. The first approach was based on genetic algorithms used in literature; the second was based on Artificial Immune Systems, and the third approach was based on a discrete implementation of Particle Swarm Optimization.

In summary, the experiments conducted show that for networks up to 40 nodes the PSO approach achieves the maximum fitness score much faster than the GA and the AIS approaches. PSO usually takes 270 iterations to reach the maximum fitness value, whereas GA needs 610 iterations and AIS needs 1420 iterations to achieve the same for a 25-node network.

Furthermore, the experiments regarding link and complete network failures showed that the approaches are able to reconfigure the networks in a reasonable amount of time. It takes a network of 20 nodes to be reconfigured by GA on average 376 ms, AIS 1035 ms, and PSO 51 ms. Again, PSO outperformed GA and AIS. For the reconfiguration of a 60-node network, GA takes 12 min, AIS 18 min, and PSO needs 21 min. Everything less than a minute is unacceptable for an automatic reconfiguration method. Of course, if sub-optimal reconfigurations are sufficient, then for example the GA method can achieve 90% after approximately 245 iterations which equates to an execution time of roughly 5.5 s; 95% can be achieved in 32 s. Therefore, the automatic reconfiguration can be used if sub-optimal configurations are sufficient.

In addition, the scalability analysis revealed that even though the performance of PSO is better than GA for smaller networks (up to 40 nodes), however, networks with a larger number of nodes are optimized faster with the GA approach as shown in Figures 6 and 7, even though better sub-optimal fitness scores can be achieved by PSO for a small number of iterations.

In conclusion, if the execution time is of essence then the PSO algorithm is the better choice achieving overlay network reconfigurations within a shorter period of time, as well as when larger network sizes are used and sub-optimal scores are acceptable. However, if the fitness is paramount, then PSO should be used for the reconfiguration of networks up to 40 nodes, and for larger networks the GA approach should be used.

Acknowledgements This research is based on work partly supported by North Dakota EPSCoR and National Science Foundation Grant EPS-0814442.

References

1. Internet Usage Statistics (2011) World Internet Users and Population Stats, last retrieved in December 2011 at http://www.internetworldstats.com/stats.htm.
2. Tarkoma S (2010) Overlay Networks: Toward Information Networking, CRC Press, Auerbach Publications, ISBN: 978-1-4398-1371-3.
3. Awerbuch B, Terzis, A (2004) A Robust Routing Algorithm for Overlay Networks, Technical Report, 2004, last retrieved in December 2011 at http://www.cs.jhu.edu/ terzis/reprouting.pdf.
4. Ludwig S.A. (2011) Nature-Inspired Reconfiguration of Overlay Networks, Proceedings of Third World Congress on Nature and Biologically Inspired Computing (NaBIC), Salamanca, Spain.
5. Kostic D, Rodriguez A, Albrecht J, Vahdat A (2003) Bullet: High bandwidth data dissemination using an overlay mesh. Proceedings of the nineteenth ACM symposium on Operating systems principles (SOSP), Bolton Landing, NY, USA.

6. Cohen B (2003) Incentives build robustness in BitTorrent, Proceedings of the 1st Workshop on Economics of Peer-to-Peer Systems (2003), Berkley, CA, USA.
7. Gummadi KP, Madhyastha H, Gribble SD, Levy HM, and Wetherall DJ (2004) Improving the reliability of internet paths with one-hop source routing, Proceedings OSDI
8. Chu YH, Rao SG, Seshan S, Zhang H (2002) A case for end system multicast, IEEE Journal of Selected Areas in Communication, 20(8).
9. Lumezanu C, Spring N, Bhattacharjee B (2006) Decentralized message ordering for publish/subscribe systems. Proceedings of the ACM/IFIP/USENIX 2006 International Conference on Middleware, Melbourne, Australia.
10. Syverson P, Goldschlag D, Reed M (1997) Anonymous connections and onion routing, Proceedings of Security and Privacy, IEEE Journal on Selected Areas in Communications, vol. 16, no. 4, pp. 482-494.
11. Jannotti J, Gifford DK, Johnson KL, Kaashoek MF, O'Toole, JW (2000) Overcast: reliable multicasting with on overlay network, Proceedings of 4th conference on Symposium on Operating System Design & Implementation - Volume 4 (OSDI'00), Vol. 4. USENIX Association, Berkeley, CA, USA.
12. Andersen D, Balakrishnan H, Kaashoek F, Morris R (2001) Resilient overlay networks, Proceedings of the eighteenth ACM symposium on Operating systems principles (SOSP '01). ACM, New York, NY, USA.
13. Lam N, Dziong Z, Mason LG (2009) Service Overlay Network Design with Reliability Constraints, Proceedings of IEEE 7th International Workshop on the Design of Reliable Communication Networks, Washington, D.C., USA.
14. Baeck T, Fogel D, and Michalewicz Z (1997) Handbook of Evolutionary Computation, IOP Publ. Ltd., Bristol, UK.
15. Caro GD, Marco Dorigo M (1998) AntNet: distributed stigmergetic control for communications networks, J. Artif. Int. Res. 9, 1 (December 1998), 317-365.
16. Montoya J, Donoso Y, Montoya E, Echeverri D (2008) Multiobjective model for multicast overlay networks over IP/MPLS using MOEA, Proceedings of International Conference on Optical Network Design and Modeling, 1-6.
17. Zhao Y, Wang J, An Y, Xia F (2009) A Layered Overlay Multicast Algorithm with PSO for Routing Web Streams, Proceedings of the International Conference on Web Information Systems and Mining (WISM '09), Wenyin Liu, Xiangfeng Luo, Fu Lee Wang, and Jingsheng Lei (Eds.). Springer-Verlag, Berlin, Heidelberg, 205-213.
18. Abraham A, Liu H, Badr Y, Grosan C (2008) A multi-swarm approach for neighbor selection in peer-to-peer networks, Proceedings of the 5th international conference on Soft computing as transdisciplinary science and technology (CSTST '08), ACM, New York, NY, USA, 178-184.
19. Montana D, Hussain T, Saxena T (2002) Adaptive reconfiguration of data networks using genetic algorithms, Proceedings of the Genetic and Evolutionary Computation Conference, pp. 11411149, San Francisco, CA, USA.
20. Koo SGM, Kannan K, Lee CSG (2006) On neighbor-selection strategy in hybrid peer-to-peer networks, Journal of Future Generation Comp. Syst. pp.732-741.
21. Ramirez AJ, Knoester DB, Cheng BHC, McKinley PK (2010) Plato: A Genetic Algorithm Approach to Run-Time Reconfiguration in Autonomic Computing Systems, Journal of Cluster Computing.
22. Keeton K, Santos C, Beyer D, Chase J, Wilkes J (2004) Designing for disasters, Proceed-ings of the 3rd USENIX Conference on File and Storage Technologies, pp. 59-62. Berkeley, CA, USA.
23. Holland JH (1975) Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor.
24. Burnet FM (1959) The Clonal selection theory of acquired immunity, Vanderbilt Univ. Press, Nashville TN.
25. de Castro LN, Timmis J (2002) An Artificial Immune Network for Multimodal Function Optimisation, Proceedings of the IEEE Congress on Evolutionary Computation, Honolulu, Hawaii, USA.

26. Kennedy J, Eberhart R (1995) Particle swarm optimization. Proceedings of IEEE International Conference on Neural Networks, Perth, Western Australia.
27. Clerc, M (2004) Discrete particle swarm optimization - illustrated by the traveling salesman problem. New Optimization Techniques in Engineering, In Studies in Fuzziness and Soft Computing, Springer