Nature-Inspired Reconfiguration of Overlay Networks

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

***Abstract*— Many different kind of overlay networking technologies have emerged over the past years. The research has focused on the development of technologies for solving the challenges to provide reliable and efficient networks for data delivery. The ability to automatically reconfigure overlay networks is essential for the future of autonomous computer systems. This paper introduces two approaches for the automatic reconfiguration of overlay networks. The first is a Genetic Algorithm approach as proposed in previous literature, and the second is a Particle Swarm Optimization approach. Both approaches are implemented in order to compare and evaluate their response to link failures and complete network failures.**

Discrete optimization; genetic algorithm; particle swarm optimization; overlay networks

I. INTRODUCTION

Many different kind of overlay networking technologies have emerged in the past years. Research and development of overlay systems have focused on developing technologies that solves the challenges of reliability and efficient data processing of networks by providing a higher-level network 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.

In particular, the enormous number of Internet users, estimated to almost 2 billions to date [1], as well as the delivery of large amounts of data and media has become commonplace. Multimedia content such as videos is posing an increasing challenge to the network, as well as the 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 networking requirements in terms of performance and reliability. Therefore, 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 application(s).
- 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 multi-purpose 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 driving 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 optimization of data mirrors is the focus. One “related work”

implementation, as well as an additional implementation is compared to measure the performance.

The remainder of this paper 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.

II. Related Work

There are several different overlay network implementation developed. One is called Overcast [4], 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 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) [5]. 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).

Nature-inspired networking techniques have also been investigated for quite some time. The research and development 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 [6]. 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 is used from ants and their indirect communication capability using pheromone.

Multi-objective evolutionary algorithms, in particular, NSGA-II was used in [7] to optimize a multicast overlay network based on two criteria; the first was to optimize the total end-to-end delay of a multicast tree, and the second was to maximize the link utilization.

A swarm-intelligence based approach called Particle Swarm Optimization was used in a layered overlay multicast approach for routing web streams [8]. 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 and Particle Swarm Optimization.

A multi-swarm approach for neighbor selection in peer-to-peer (P2P) overlay networks is described in [9]. Their approach is inspired by the communalities 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 [10] genetic algorithms are applied to reconfigure the topology and link capacities of an operational network. It does this in order to adapt to changes in its operation 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 [11]. The strategy enhances the decision process performed for transfer coordination. An investigation of the strategy revealed that it affects the system throughput and distribution efficiency as well as peer contribution, in particular for low-connectivity peers.

Plato is a genetic algorithm approach to run-time reconfiguration [12]. 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 tradeoffs 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, and demonstrates 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, in this paper the fitness functions are normalized in order to compare the three different measures and to provide a fitness score in the range between zero and one. Also, besides the use of genetic algorithms, an algorithm based on Particle Swarm Optimization is implemented and evaluated. As can be seen in the evaluation section, the additionally implemented algorithm shows a better performance than the Genetic algorithm approach.

III. 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 [13].

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 [13].

There are two types of remote mirroring protocols that are synchronous or 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 batch-style.

TABLE I. LINK PROPAGATION METHODS

Time interval	Avg. data batch size in GB
0	0
1 min	0.0436
5 mins	0.2067
1 hr	2.091
4 hrs	6.595
12 hrs	15.12
24 hrs	27.388

The optimization design criteria are the same as for Plato [12]. 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 exceed 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.

Two 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 [12]; the second one is an implementation of the discrete Particle Swarm Optimization approach.

The fitness function used as a measure for the two algorithms are slightly modified compared to the Plato approach [12]. 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. The fitness function consist 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)).

$$Fit_{overall} = w_1 * Fit_{cost} + w_2 * Fit_{perf} + w_3 * Fit_{rel} \quad (1)$$

$$\sum_{i=1}^{3} w_i = 1 \quad (2)$$

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

$$F_{cost} = 1 - \frac{cost}{budget} \quad (3)$$

where *cost* is the sum of operational expenses of all active links and *budget* is a user supplied value (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 measured over all links in the underlying network; and $bandwidth_{sys}$ is the total available bandwidth across the overlay network given by 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:

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

where $links_{used}$ is the number of active links, and latency over all active links, and $links_{\max}$ is the maximum number of possible links given by 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.

A. Genetic Algorithm Approach

Genetic algorithms (GA) [14] are a class of stochastic search algorithms based on biological evolution. In particular, the principles of 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.

TABLE II. GA PARAMETERS

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

Table 2 lists the parameters used for the implementation, which are identical to the ones proposed in Ramirez et al. [12] with the exception of the normalized fitness function as

outlined earlier. Two-point crossover is employed, as well as Tournament selection is used.

B. Particle Swarm Optimization Approach

Particle Swarm Optimization (PSO) as introduced in [15], 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. PSO's basic equations are:

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

$$v_{ij}(t+1) = w(t)v_{ij}(t) + c_1 r_{1j}(t)\left(xBest_{ij}(t) - x_{ij}(t)\right) + c_2 r_{2j}(t)\left(xGBest_j(t) - x_{ij}(t)\right) \quad (7)$$

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 [16]. 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 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 with a certain probability.

TABLE III. 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

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 3 shows the specific parameters chosen for the implementation.

IV. EXPERIMENTS AND RESULTS

The two algorithms were tuned with the parameter values as given in the previous section. Since the GA and PSO implementation involve probabilities, 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.

TABLE IV. FITNESS FUNCTION AND FITNESS VALUES (500 ITERATIONS)

Fitness function	w_1	w_2	w_3	Best fitness - GA	Best fitness – PSO
F1	1/3	1/3	1/3	0.82331	0.83333
F2	1	0	0	0.89102	0.91290
F3	0	1	0	0.89653	0.90872
F4	0	0	1	0.60379	0.60862
F5	0.5	0.5	0	0.89786	0.89995
F6	0	0.5	0.5	0.73942	0.75000
F7	0.5	0	0.5	0.73018	0.74992

The first set of experiments was performed measuring the accuracy of both 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.

Table 4 shows the results of the fitness scores of both approaches 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. The worst fitness score is observed for F4, when optimizing the overlay network based on reliability.

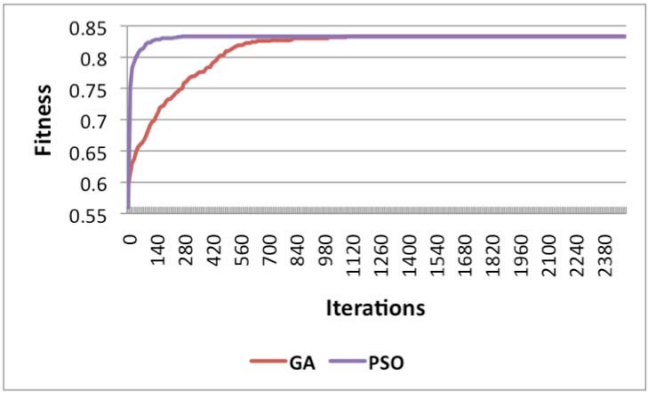

Figure 1. Fitness of GA and PSO approach.

Figure 1 shows the fitness values plotted for increasing iterations. It can be observed that the PSO approach reaches the maximum fitness score of 0.83333 at iteration 270, which is much earlier than the GA approach, which needs 1,100 iterations. The fitness function F1 with an equal weight distribution was used.

A. *Investigation of Link Failures*

Figure 2 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. 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 automatic reconfiguration needs to be restarted.

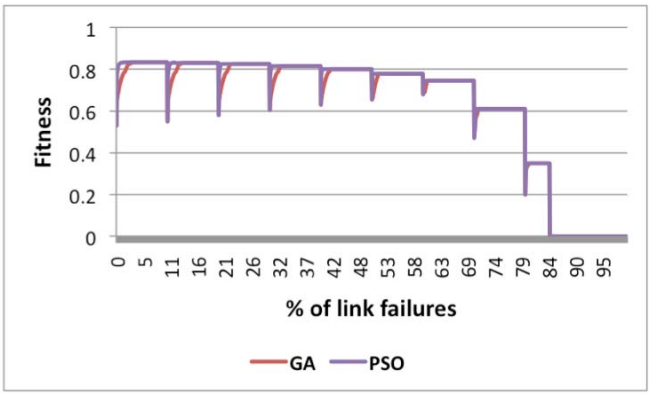

Figure 2. Fitness scores with increasing percentage of link failures.

B. *Investigation of Network Failures*

Figure 3 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, both approaches, GA and PSO, can reconfigure overlay networks and achieve the maximum fitness score after around 1,200 and 300 iterations respectively. As expected, the trend of both curves is as in Figure 1.

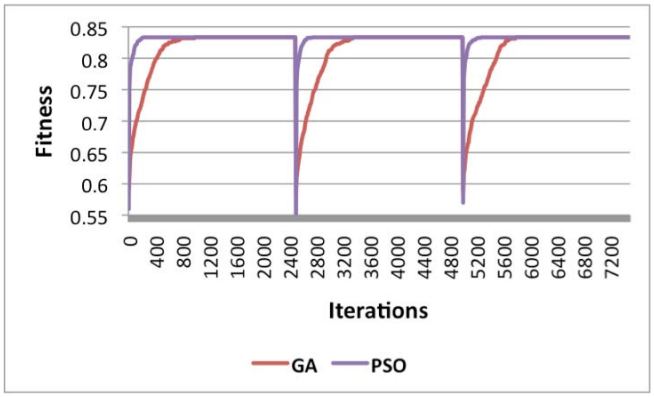

Figure 3. Reconfiguration of complete overlay network with 25 nodes.

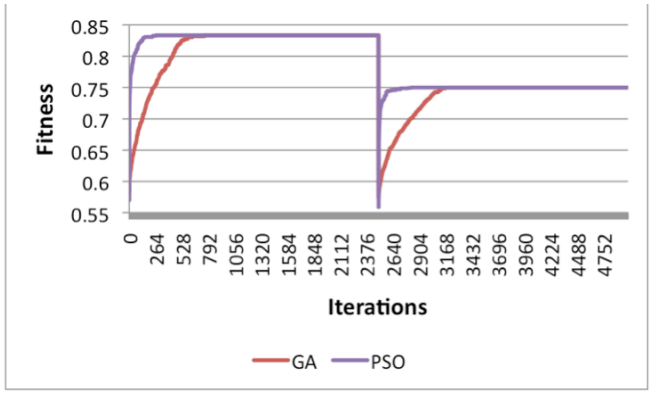

Figure 4. Reconfiguration of 25-node network with varying weight distribution of fitness function.

Figure 4 shows the reconfiguration of a 25-node network. The first evolution shows the fitness trend when F1 is used (weights are all 1/3). 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.

V. CONCLUSION

This paper investigated two approaches for the reconfiguration of overlay networks and the experimental results could be used as a guideline for overlay network construction and configuration. The first approach was based on genetic algorithms as used in literature; the second approach was based on a discrete implementation of Particle Swarm Optimization.

In summary, the experiments conducted show that for networks of 25 nodes the PSO approach achieves the maximum fitness score much faster than the GA approach. PSO usually takes 270 iterations to reach the maximum fitness value, whereas GA needs 1,100 iterations to achieve the same for a 25-node network. Furthermore, the experiments regarding link and complete network failures showed that both 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 and PSO 51 ms. Again, PSO outperformed GA.

In conclusion, the PSO algorithm outperforms the GA algorithm when reconfiguring overlay networks. However, a scalability analysis needs to be conducted in order to find out which approach will scale better with increasing network sizes. Besides this, future work will investigate and implement another evolutionary method based on Artificial Immune Systems in order to examine the suitability in the area of overlay networks.

ACKNOWLEDGMENT

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

REFERENCES

[1] Internet Usage Statistics, World Internet Users and Population Stats, last retrieved in May 2011 at http://www.internetworldstats.com/stats.htm.

[2] S. Tarkoma, Overlay Networks: Toward Information Networking, CRC Press, Auerbach Publications, ISBN: 978-1-4398-1371-3, 2010.

[3] Baruch Awerbuch, Andreas Terzis, A Robust Routing Algorithm for Overlay Networks, Technical Report, last retrieved at http://www.cs.jhu.edu/~terzis/reprouting.pdf, 2004.

[4] J. Jannotti, D.K. Gifford, K.L. Johnson, M. Frans Kaashoek, and J.W. O'Toole. 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, 14-14, 2000.

[5] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris. Resilient overlay networks. In Proceedings of the eighteenth ACM symposium on Operating systems principles (SOSP '01). ACM, New York, NY, USA, 131-145, 2001.

[6] G. Di Caro and M. Dorigo. AntNet: distributed stigmergetic control for communications networks. *J. Artif. Int. Res.* 9, 1 (December 1998), 317-365, 1998.

[7] J. Montoya, Y. Donoso, E. Montoya, and D Echeverri, Multiobjective model for multicast overlay networks over IP/MPLS using MOEA. *2008 International Conference on Optical Network Design and Modeling*, 1-6, 2008.

[8] Y. Zhao, J. Wang, Y. An, and F. Xia. A Layered Overlay Multicast Algorithm with PSO for Routing Web Streams. In *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, 2009.

[9] A. Abraham, H. Liu, Y. Badr, and C. Grosan. A multi-swarm approach for neighbor selection in peer-to-peer networks. In *Proceedings of the 5th international conference on Soft computing as transdisciplinary science and technology* (CSTST '08). ACM, New York, NY, USA, 178-184, 2008.

[10] D. Montana, T. Hussain, T. Saxena: Adaptive reconfiguration of data networks using genetic algorithms. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 1141–1149, San Francisco, CA, USA, 2002.

[11] S.G.M. Koo, K. Kannan, C.S.G. Lee, On neighbor-selection strategy in hybrid peer-to-peer networks, Journal of Future Generation Comp. Syst. pp. 732-741, 2006.

[12] A.J. Ramirez, D.B. Knoester, B.H.C. Cheng, and P.K. McKinley. Plato: A Genetic Algorithm Approach to Run-Time Reconfiguration in Autonomic Computing Systems. Journal of Cluster Computing, 2010.

[13] K. Keeton, C. Santos, D. Beyer, J. Chase, J. Wilkes: Designing for disasters. Proceedings of the 3rd USENIX Conference on File and Storage Technologies, pp. 59-62. Berkeley, CA, USA, 2004.

[14] J.H. Holland: Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor, 1975.

[15] J. Kennedy, R. Eberhart: Particle swarm optimization. Proceedings of IEEE International Conference on Neural Networks, 1995.

[16] M. Clerc: Discrete particle swarm optimization - illustrated by the traveling salesman problem. New Optimization Techniques in Engineering, Springer, 2004.