SERBIAN JOURNAL OF ELECTRICAL ENGINEERING
Vol. 6, No. 1, May 2009, 43-55
UDK: 004.722.2:519.712

Using of Coevolutionary Algorithm on P2P Networks

Alireza Rezaee[1]

Abstract: Multicast routing is the basic demand to provide QOS (Quality of service) in multimedia streaming on peer to peer networks. Making multicast trees optimizing their delay cost and considering nodal and links limited bandwidth (load balance constraints) is a NP-hard (Nondeterministic Polynomial time hard) problem. In this paper we have used Co-evolutionary Algorithm to make multicast trees with optimized average delay from source to the clients considering the limited capacity of nodes and links in application layer. The numeric results obtained are shown that the costs have been much improved comparing with other existent non-GA (Genetic Algorithm) approaches. Also we have used only a portion of every nodal outage degree and this has improved the results comparing to use of the entire outage degree.

Keywords: Co-evolutionary algorithm, P2P networks, Multicast tree.

1 Introduction

Today multimedia streaming over peer to peer networks and internet has been increased and existence of a lot of applications like video on demand and video conferencing put critical importance on providing QOS over them. The most important approach to provide this and reduce the load of the network is to make multicast trees rooted in the source spanning all the nodes applying the data. Making multicast trees in the networks considering the large number of applying nodes and huge number of limiting parameters like constrained links bandwidths and limited capacity of transit nodes has converted the problem to a NP-hard one and can be considered in various network layers.

Many approaches have been introduced by now to produce the multicast trees based on non-Genetic algorithm solutions Banerjee et.al. in OMNI [1] tried to find the optimized tree by first performing an initialization step and then improving it by time Their approach was to optimize $\sum_i c_i T_{i,v}$ when $T_{i,r}$ is the delay between transit node *i* and the root averaged on c_i the number of clients expected to receive data from that node. They used the entire permitted nodal degree in their paper. But this could approach only a nearly optimized tree during the sending time.

[1]Islamic Azad University-Hashtgerd Branch, Karaj, Iran; E-mail: arrezaee@yahoo.com

Practical experiments show that when the number of nodes and the number of constrained parameters are large Genetic algorithm approaches are capable of capturing the best optimized trees [2,3].

In this paper we have used Co-evolutionary algorithm to provide multicast trees rooted in source node minimizing average delay cost mentioned for OMNI [3] but with more flexibility in selecting the number of copies each node produces.

Also we have not used all outage nodal degrees and permitted our algorithm to use only some of them and this had improved the final results comparing with former solutions.

The rest of this paper is organized as follows. The multicast routing problem its parameters and some other approaches to it are described in Section 2**.** In Section 3 Genetic algorithm and Co-evolutionary GA approaches and their fundamentals are described and our approach, model, parameters and the details of Co-evolutionary we have used is explained in Section 4. At last the numeric results obtained are shown in Section 5 and they are compared with other approaches and a conclusion is derived in Section 6.

2 Multicast Problem

Multicasting is the simultaneous transmission of data to several applicant nodes through a tree which provides two main advantages. First the data can be transmitted in parallel to the various destination nodes along the tree branches and secondly it must be copied only when it is required and this will reduce the load of the network and probability of congestion occurrence comparing with sending data directly to all receivers. Fig. 1 shows a multicast tree, data is copied only in branches to be transmitted through them. So in multimedia streaming when real-time demands like least delay and jitter arise, making optimized multicast trees will be of great importance.

Optimizing these trees can be considered in network layer (Network Layer Multicasting) or application layer (Application Layer Multicasting) [4]. Some NLM (Network Layer Multicasting) problems are link load balance which considers the transit links bandwidths along the tree branches if they can transit the data required, the destination cost which is the delay experienced by the destination nodes [5] and the occurrence of nodes and links in several branches of the tree which leads to re-transiting the repeated data in network layer and over load the network. Some ALM (Application Layer Multicast) problems are the diameter of the tree, the total delay in the application layer, nodal load balance which considers the maximum degree of nodes out streaming that is limited by their outage bandwidth, copying capability and links connected to them [4].

Banerjee et.al. in OMNI [3] at first performed a decentralized initialized step in root and made an initial tree according to the direct distance of every node to the root. They used the entire permitted degree of every node in their approach and then performed some local periodic transformation on the tree to improve (minimize) the total average-latency of the degree-bounded spanning tree. Some of these transformations were for example child promote, parent-child swap, Iso-level-2 swaps [6]. Some of these transformations are shown in Fig. 1. The other transformations were probabilistic ones performed by a low probability and all were performed only if Λ_i of the root of the involved sub-tree decreased.

$$\Lambda_i = \left\{ \begin{array}{ll} 0, & \text{if } i \text{ is a leaf } MSN \\ \sum_{j \in \text{Children}(i)} s_j l_{i,j} + \Lambda_j, & \text{otherwise} \end{array} \right\}$$

This approach is able to reach only nearly-optimized trees because of using the entire degree of nodes in initial trees and weaker solution compared by GA and we showed that our algorithm (detailed in Section 4) based on Co-evolutionary GA using not all the permitted degree approaches better solutions.

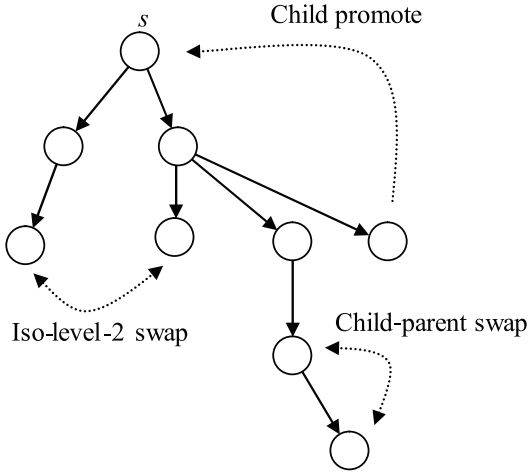


Fig. 1 – *A Multicast tree and some transformation in OMNI is shown: copying is done in the branches.*

Peng et.al. in [4] used Genetic algorithm to make multicast trees but they consider NLM problems such links and nodes load balance more and they didn't make target to optimize the total averaged delay that is very important in multimedia streaming.

Our approach using Co-evolutionary is based on optimizing the averaged delay considering the nodal load balances in the application and network layer.

3 Genetic Algorithm and Co-Evolutionary in Summary

Genetic algorithm (GA) is a heuristic used to find approximate solutions to difficult-to-solve problems through application of the principles of evolutionary biology to computer science. Genetic algorithms use biologically-derived techniques such as inheritance, mutation, natural selection, and recombination (or crossover). They are a particular class of evolutionary algorithms and are typically implemented as a computer simulation in which a population of abstract representations (called chromosomes) of candidate solutions (called individuals) to an optimization problem evolves toward better solutions. Traditionally, solutions are represented in binary as strings of 0s and 1s, but different encodings are also possible for example in our approach real value chromosomes is preferred. The evolution starts from a population of completely random individuals and happens in generations. In each generation, the fitness of the whole population is evaluated, multiple individuals are stochastically selected from the current population (based on their fitness), modified (mutated or recombined) to form a new population, which becomes current in the next iteration of the algorithm .

Two elements are required for any problem before a genetic algorithm can be used to search for a solution: First, there must be a method of representing a solution in a manner that can be manipulated by the algorithm. Traditionally, a solution can be represented by a string of bits, numbers or characters. Second, there must be some method of measuring the quality of any proposed solution, using a fitness function.

Initially many individual solutions are randomly generated to form an initial population. The population size depends on the nature of the problem. Traditionally, the population is generated randomly, covering the entire range of possible solutions (the search space). Occasionally, the solutions may be “seeded” in areas where optimal solutions are likely to be found.

During each successive epoch, a proportion of the existing population is selected to breed a new generation. Individual solutions are selected through a fitness-based process, where fitter solutions (as measured by a fitness function) are typically more likely to be selected. Certain selection methods rate the fitness of each solution and preferentially select the best solutions. Other methods rate only a random sample of the population, as this process may be very time-consuming.

Most functions are stochastic and designed so that a small proportion of less fit solutions are selected. This helps keep the diversity of the population large, preventing premature convergence on poor solutions. Popular and well-studied selection methods include roulette wheel selection and tournament selection.

The next step is to generate a second generation population of solutions from those selected through genetic operators: crossover (or recombination), and mutation.

For each new solution to be produced, a pair of “parent” solutions is selected for breeding from the pool selected previously. By producing a "child" solution using the above methods of crossover and mutation, a new solution is created which typically shares many of the characteristics of its “parents.” New parents are selected for each child, and the process continues until a new population of solutions of appropriate size is generated.

By crossover every two parents generate two children containing a combination of their parent’s characteristics and by mutation one parent produces a child with some conversions.

These processes ultimately result in the next generation population of chromosomes that is different from the initial generation. Generally the average fitness will have increased by this procedure for the population, since only the best organisms from the first generation are selected for breeding.

This generational process is repeated until a termination condition has been reached. Common terminating conditions are: A solution is found that satisfies minimum criteria, Fixed number of generations reached, Allocated budget (computation time/money) reached, The highest ranking solution's fitness is reaching or has reached a plateau such that successive iterations no longer produce better results, Manual inspection, Combinations of the above.

In Co-evolutionary two parallel Genetic algorithms are run together and parameters to be found divided in two separate segments are sent among them. Each GA evaluates its generation by it self’s parameters in the generation using some number of parameters belonging to the other group and averaging on them. The new generation is evaluated and the results are transmitted among the genetic algorithms till the algorithm is terminated. Co-evolutionary genetic algorithms optimize more rapidly compared by simple genetic algorithms and they are used when the number of parameters is large [1,2]. Co-evolutionary algorithm is schematically shown in Fig. 2.

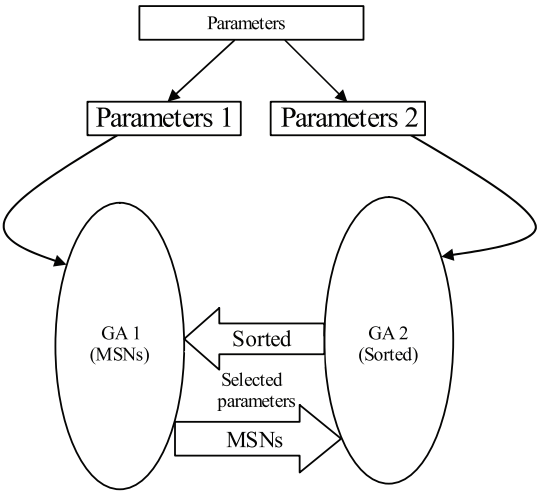


Fig. 2 – *Co-Evolutionary Genetic algorithm is shown schematically.*

4 Our Co-Evolutionary Based Approach

A. Network Model:

Our network model is an application layer model which is obtained from a network layer model. It can be summarized az MeshNet(N,P,c), mesh MeshNet is consisted from n nodes ($n_i \subseteq N$) node n_1 as the root, the paths $P \subseteq NxN$ between them and a cost $c_{i,j}$ is associated with every $p_{i,j} \subseteq P$ describing the delay of that path. Every node n_i has a maximum permitted degree (d_i) that is its maximum permitted children and a number of clients attached to it (c_i), Fig. 3.

B. Our Approach:

We have shown every tree by two main describing arrays: sorted and MSNs (Maximum Signal-to-Noise ratio). Sorted array is an array containing n elements each number of a node which describes the order that the nodes have been used to be children of the formers. In every sorted array the first element is obviously 1. The second array is MSNs, ith element containing the number of children the node n_i has in the tree. It is obvious that MSN(*i*) is between 1 and d_i. The trees are shown by *N*×*N* matrices which the parents and children of every node is determined in front of it.

Our tree maker walks on the sorted array and selects each node that is not a parent and according to its capacity in MSNs array chooses the nodes which are not children yet from the sorted and so on.

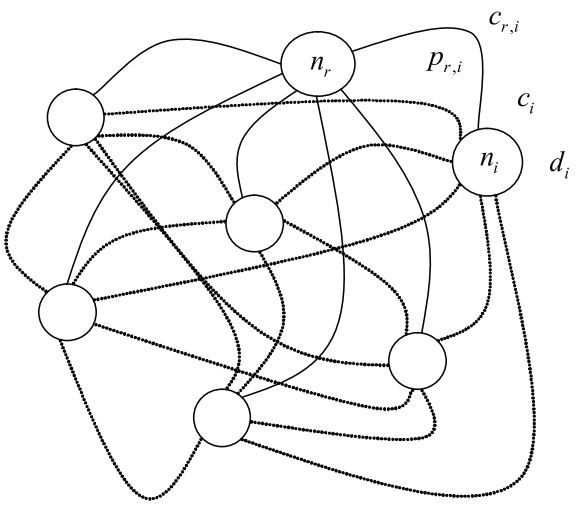


Fig. 3 – *The network model used in this paper.*

MSNs array chooses the nodes which are not children yet from the sorted and so on.

We have used a Co-evolutionary algorithm running two GAs parallel and separated the parameters as the MSNs and the sorted.

The initial population in our GA is changed during the experiments. Our algorithm generates an initial population of trees by randomly generating sorted (random permutations of n with first element 1) and MSNs arrays. But always the first chromosome of the initial population is the initial tree made by OMNI [3]. Then a number of parents (changing parameters) are chosen and 4 operations are done to make children for next population:

Crossover: we have two kinds of crossovers on the sorted arrays, in the first one, one of parent's sorted array determines the order of choosing elements from the other parent's sorted to produce one of children's sorted and vise versa. In the second kind every element of a child is chosen from one parent and the next from the other parent and so on. The MSNs are simply combined from the middle element.

Mutation: Two random elements of the sorted array are swapped and two random elements of MSNs array are randomly chosen between 1 and the maximum permitted degree.

Immigration: Some numbers of the population are died and instead the new random people are immigrated by randomly chosen sorted and MSNs array.

Genocide: This operation is done when the algorithm is trapped in local minimums to rescue from searching that area of answer space. In this operation

a number of this population are converted intelligently just to change their position in the answer space. To do this every sorted is rearranged in opposite order it was before. It means that we order the stored from end to the beginning. Every MSNs element is decreased from the max value and the new number is replaced. This operation is done when the best cost of the best tree has been unchanged in several (changing parameter) generations.

The multicast trees are evaluated by the average tree latency used in OMNI and the bests are captured.

In our Co-evolutionary genetic algorithm one GA is run to optimize the sorted array while receiving the MSNs from the other GA and vise versa. Every GA uses the best 3 answers of the other one to evaluate its parameters.

We have changed the parameters of our operations during one experiment if the algorithm is trapped in a local minimum for several numbers of generations and continue our algorithm till the max number of generations is achieved.

Also we have innovated not to use all the outage nodal degree that every node is permitted to use (as is done in OMNI) and permitted our algorithm to use every portion of this degree. This has improved the results very much as is shown in the simulation experiments section.

5 Simulation Experiments

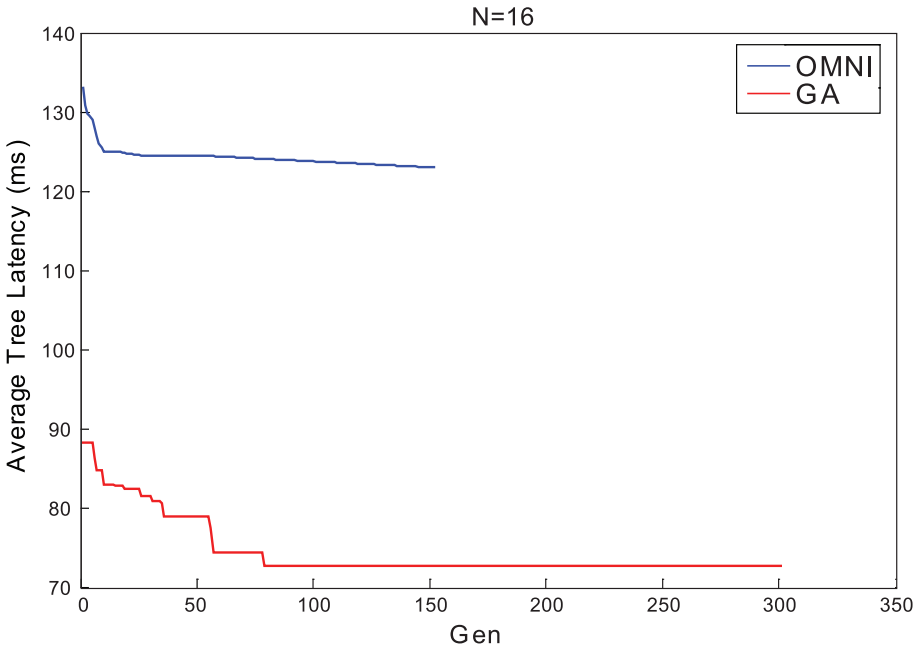
We have run our algorithm in a network modeled as described in Section 4**.** The total nodes are 16 and 256 and the max number of clients a node can accept is 10. The average outage degree of nodes is 4 and delays between nodes are distributed between 0 and 200ms.

We have run our simple and co-evolutionary genetic algorithm and compared with the previous approaches. The results show that genetic algorithms (Co-evolutionary and simple) approach very better results with less delay values compared by other solutions and co-evolutionary algorithms approach the answer so much sooner than simple genetic ones.

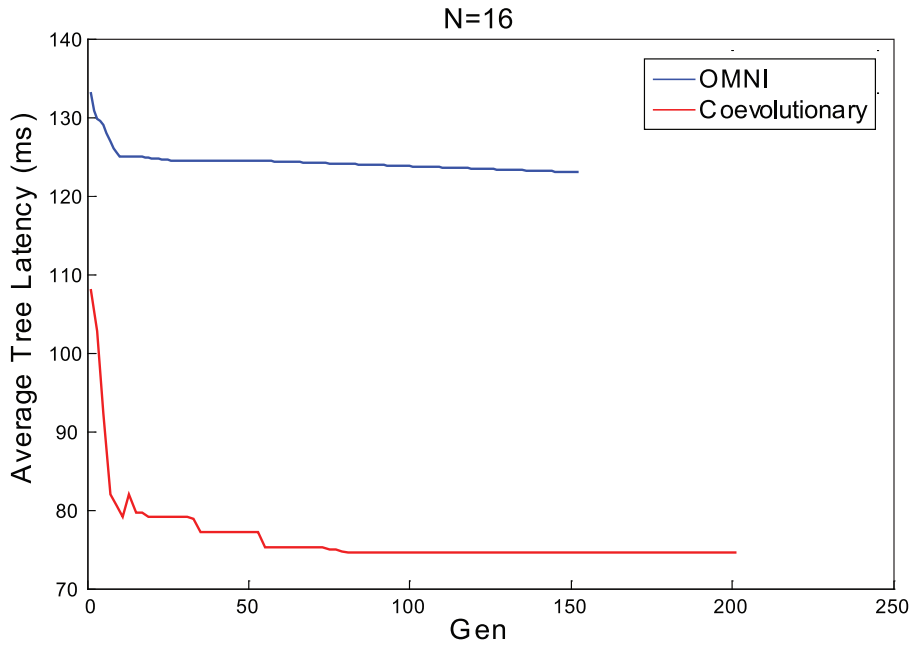
As is shown in Figs. 5a and 5b simple and Co-evolutionary genetic algorithms approach very better results compared with the others specially in few number of nodes. When the number of nodes grows algorithms approach better results but not as fast as when nodes are fewer.

Fig. 5c shows how much sooner Co-evolutionary algorithms approach the answer comparing simple genetic algorithms and they get answers which simple GA obtains many generations later.

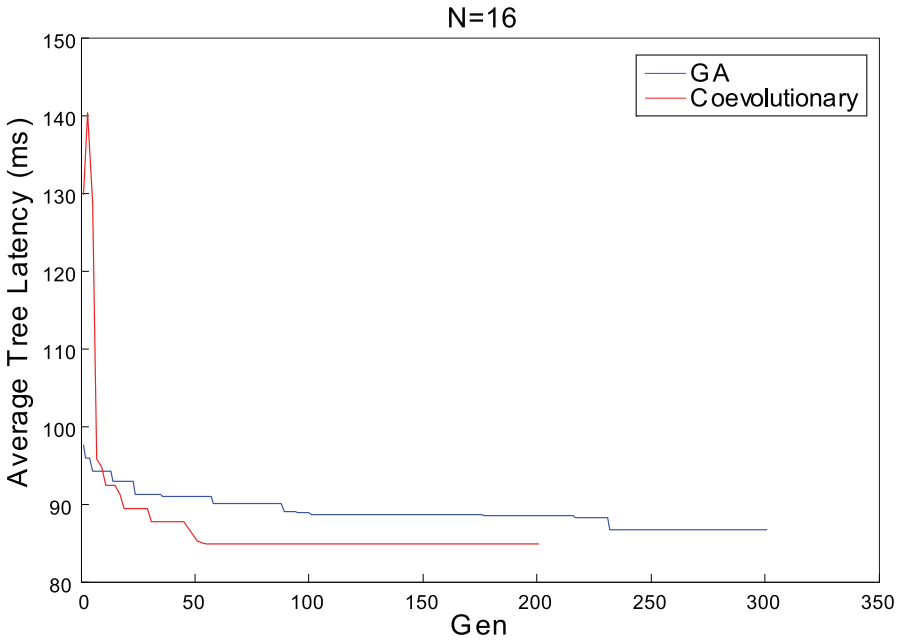
The large distance between omni results and our results shows that our innovation in not filling every MSN by all its degree has improved the results very much.



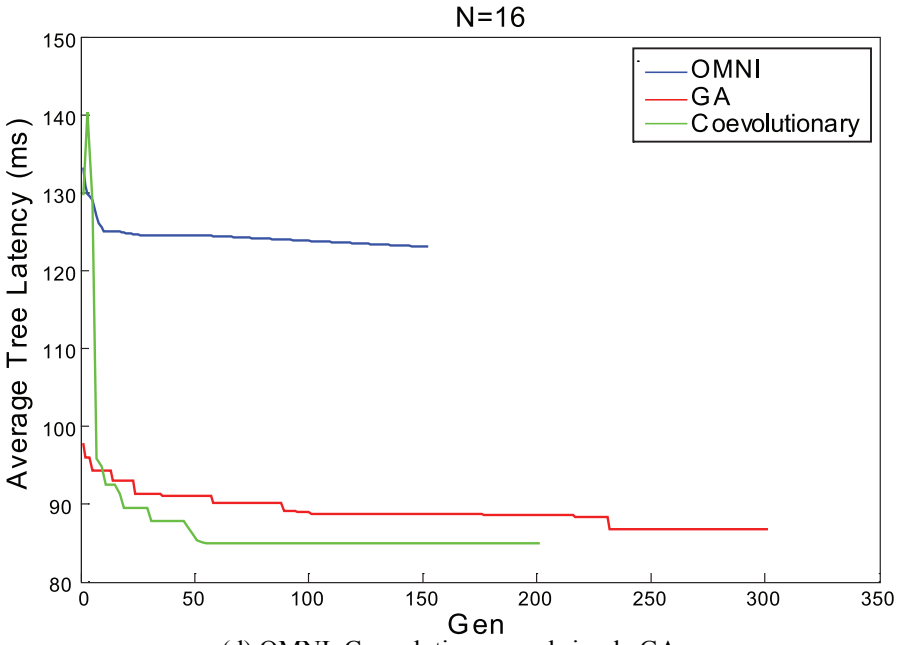
(a) Omni compared with simple GA.



(b) Omni compared with Coevolutionary.

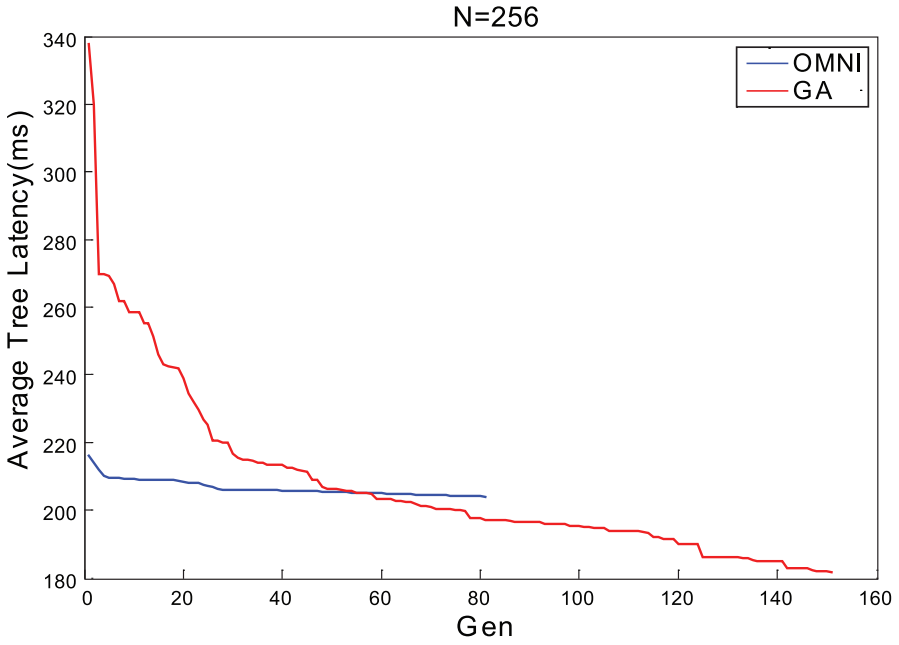


(c) Coevolutionary compared with GA

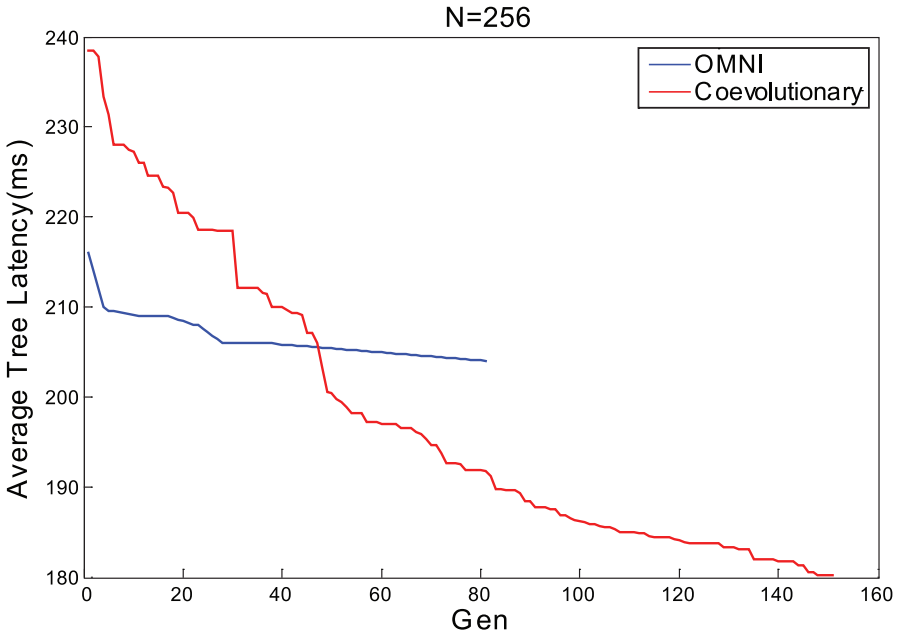


(d) OMNI, Coevolutionary and simple GA

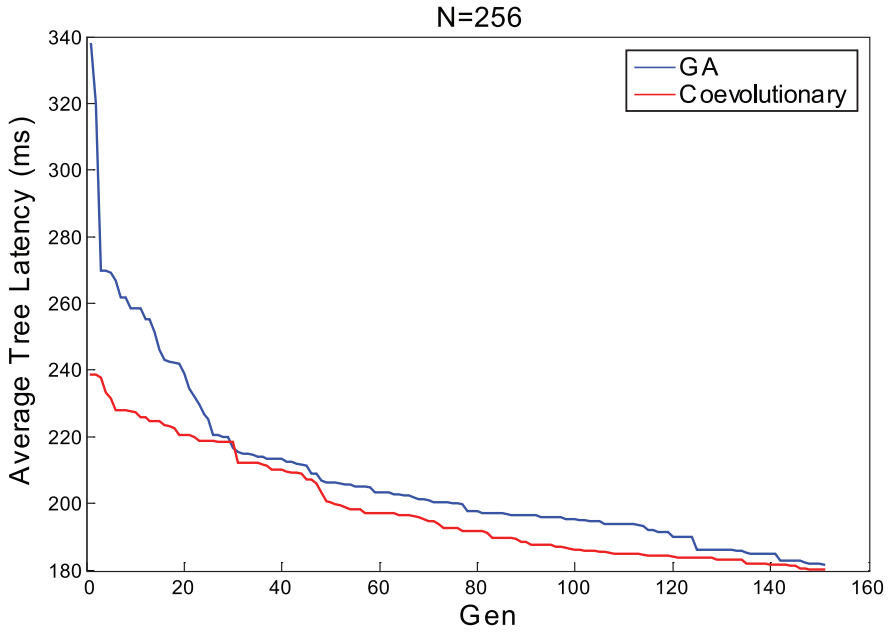
Figs. 4a, 4b, 4c, 4d – *OMNI, GA and Co-evolutionary are compared* $(N = 16)$.



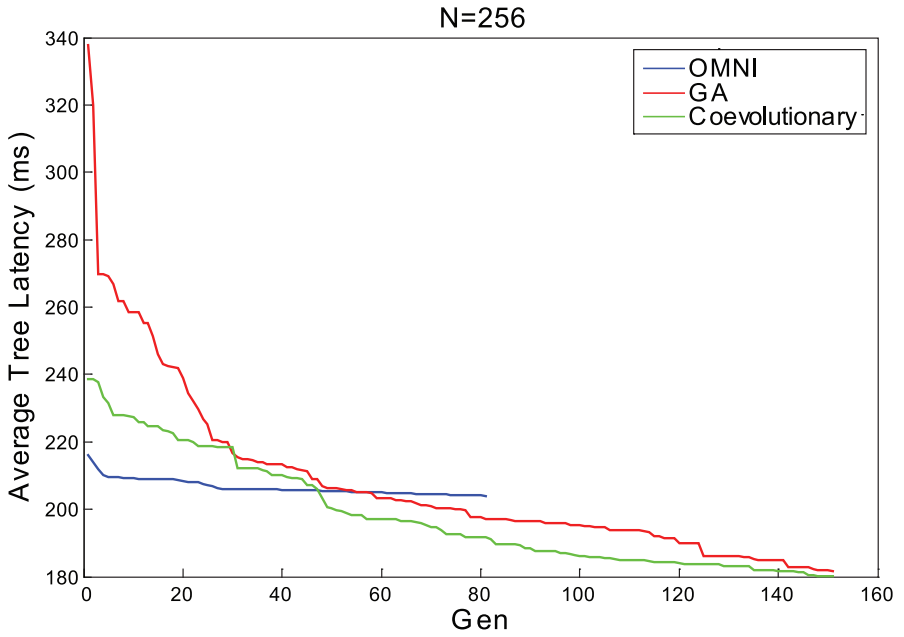
(a) Omni compared with simple GA.



(b) Omni compared with Coevolutionary.



(c) Coevolutionary compared with GA.



(d) OMNI, Coevolutionary and simple GA.

Figs. 5a, 5b, 5c, 5d – *OMNI, GA and Co-evolutionary are compared* ($N = 256$).

6 Conclusion

We have used Co-evolutionary algorithm to make multicast trees with minimum average tree latencies satisfying the outage node degrees and link load balance constraints.

According to the numeric results shown in the figures in Section 5 it can be observed that our method approaches multicast trees with much less average tree latencies compared with previous approaches like OMNI. Using co-evolutionary algorithms we approach the optimized tree much sooner than simple algorithms.

The other important technique we have used is not to use all outage nodal degrees and permit our algorithm to select this number and this has improved the results very much.

In this paper we have only considered ALM problems, it appears the most proper trees can be achieved by considering network layer problems and make trees using less nodes and links (more shared ones between various paths on the application layer tree) in network layer.

7 References

[1] S. Banerjee, C. Kommareddy, K. Kar, B. Bhattacharjee, S. Khuller: Construction of an Efficient Overlay Multicast Infrastructure for Real-time Applications, INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies, Vol. 2, 2003, pp. 1521-1531.

[2] N. Chaiyaratana, A.M.S. Zalzala: Recent Developments in Evolutionary and Genetic Algorithms: Theory and Applications, Genetic Algorithms In Engineering Systems: Innovations And Applications GALESIA 97, Second International Conference On (Conf. Publ. No. 446), 1997, pp. 270-277.

[3] H. Handa, N. Baba, O. Katai, T. Sawaragi, T. Horiuchi: Genetic Algorithm Involving Coevolution Mechanism to Search for Effective Genetic Information, IEEE International Conference on Evolutionary Computation, 1997, pp. 709-714.

[4] C. Peng, D. Qionghai, W. Qiufeng: An Application Layer Multicast Routing Algorithm Based on Genetic Algorithms, Proceedings of the 8th International Conference on Telecommunications ConTEL 2005, Vol. 2, 2005, pp. 413-418.

[5] S. Ahn, D.H.C. Du: A multicast Tree Algorithm Considering Maximum Delay Bound for Real-time Applications, Proceedings of 21st IEEE Conference on Local Computer Networks, 1996, pp. 172-181.

[6] S. Mao, X. Cheng, Y.T. Hou, H.D. Sherali: Multiple Description Video Multicast in Wireless Ad Hoc Networks, Proceedings of First International Conference on Broadband Networks, 2004, pp. 671-680.