Cumulative Updating of Network Reliability with Diameter Constraint and Network Topology Optimization

Denis A. Migov(✉), Kseniya A. Nechunaeva,
Sergei N. Nesterov, and Alexey S. Rodionov

Institute of Computational Mathematics and Mathematical Geophysics SB RAS,
prospect Akademika Lavrentjeva 6, 630090 Novosibirsk, Russia
mdinka@rav.sscc.ru, ksu.nech@gmail.com,
cepera_666@inbox.ru, alrod@sscc.ru
http://www.sscc.ru/

Abstract. Reliability-based optimization of a network topology is to maximize the network reliability within certain constraints. For modeling of unrelaible networks we use random graphs due to their good applicability, wide facilities and profound elaborating. However, graph optimization problems in conditions of different constraints are NP-hard problems mostly. These problems can be effectively solved by optimization methods based on biological processes, such as genetic algorithms or clonal selection algorithms. As a rule, these techiques can provide an applicable solution for network topology optimization within an acceptable time. In order to speed up fitness function calculation, we improve operators of a genetic algorithm and a clonal selection algorithm by using the method of cumulative updating of lower and upper bounds of network reliability with diameter constraint. This method allows us to make a decision about the network reliability (or unreliability) with respect to a given threshold without performing the exhaustive calculation. Based on this method, we obtain the genetic algorithm and the clonal selection algorithm for network topology optimization. Some computational results are also presented for demonstration of an applicability of the proposed approach.

Keywords: Network reliability · Network topology optimization · Genetic algorithm · Clonal selection algorithm · Random graph · Diameter constraint · Factoring method · Cumulative updating

1 Introduction

Genetic algorithms (GAs) and clonal selection algorithms (CSAs) [1–3] are widely used for network topology optimization [4–6]. These approaches are based on selection and recombination of promising solutions. GAs and CSAs

Supported by Russian Foundation for Basic Research under 16-37-00345 grant.

O. Gervasi et al. (Eds.): ICCSA 2016, Part I, LNCS 9786, pp. 141–152, 2016.
DOI: 10.1007/978-3-319-42085-1_11

have achieved great success in solving numerous graph optimization problems. However, the performance of these algorithms depends on the right choice of operators (such as selection, crossover, cloning, mutation etc.), and method for calculation of fitness function. Otherwise, computational time of GAs and CSAs can be increased on an enormous scale. On the other hand, we can significantly improve the algorithms performance by using various techniques for rejecting inapplicable chromosomes.

In present paper we deal with the problem of obtaining the most reliable network topology within a given budget.

It is assumed that network has unreliable elements which are subject to random fault that occur mutually independently. Random graphs are commonly used for modeling such networks. We consider the case of absolutely reliable nodes and unreliable edges which corresponds to real networks where the reliability of nodes is much higher than reliability of communication links.

One of the basic reliability measures for such networks is the probabilistic connectivity, i.e. the probability of a given subset of nodes to be connected. This measure is quite well examined, various exact and approximate reliability calculation methods have been proposed [7]. Another popular measure of network reliability is the diameter constrained network reliability (Petingi and Cancela, 2001 [8,9]). Further on we use abbreviation DCNR for notation of diameter constrained network reliability. DCNR is a probability that every two nodes from a given set of terminals are connected with a path of length less or equal to a given integer. By the length of a path we understand the number of edges in this path. This reliability measure is more applicable in practice, for example, in the case of P2P networks [10]. However, the problems of computing these characteristics are known to be NP-hard. Moreover, DCNR calculation problem is NP-hard for most combinations of a diameter value and a number of terminals [11].

The new approach in the area of network reliability analysis was introduced in [12,13]: cumulative updating of lower and upper bounds of all-terminal network reliability for faster feasibility decision. This method allows to decide the feasibility of a given network without performing the exhaustive calculation. The approach was further developed with help of network decomposition [14]. In our previous research [15] we've proposed the method for network topology optimization with use of cumulative updating. The most reliable topology was sought from the point of view of all-terminal reliability. In present study we obtain the cumulative updating method for DCNR and use it for network topology optimization.

2 Problem Statement

Let us have a set of vertices $V = \{V_1, ..., V_n\}$ and a set $S = \{S_1, ..., S_t\}$ of weighted edges, $C = \{C_1, ..., C_t\}$ and $P = \{r_1, ..., r_t\}$ — weights and connection probabilities of edges from S accordingly; $K = \{k_1, ..., k_l\}$ — terminal nodes. Values of budget constraint C^* and diameter constraint d are given. We use notation $R_K^d(G)$ for reliability of G with diameter constraint d. Probabilistic connectivity of G is denoted by $R(G)$. For rigorous definitions of the described reliabilities measures we refer to [16] (for probabilistic connectivity) and [17] (for DCNR).

Need to construct connected undirected graph $G = (V, E \subset S)$ with maximum diameter constrained reliability value with following restrictions:

$$\begin{cases} R_k^d(G) \to \max; \\ Weight(G) < C^*. \end{cases} \tag{1}$$

Preference is given to the cheapest solution in case of equal probability values.

3 Brief Survey of Exact Methods for Network Reliability Calculation

Present section describes some methods of network reliability calculation, but this is not meant to be a complete summary of the work in this field.

The usually used method for calculating any network reliability measure is the factoring method. The main idea is to partition the probability space into two sets based on the success or failure of a chosen network's element which is referred to as factored element. Thus, given a graph G and a factored element e we will obtain two graphs G/e and $G\backslash e$. In the first of them the factored element is absolutely reliable and in the second one the factored element is absolutely unreliable, e.g. it could just be removed. The probability of G/e is equal to the reliability of factored element and the probability of $G\backslash e$ is equal to the failure probability of factored element. The same procedure is to be applied for the both graphs involved. Using the total probability law the following expression is obtained [7]:

$$Rel(G) = r_e Rel(G/e) + (1 - r_e) Rel(G\backslash e), \tag{2}$$

Recursions continue until either obtained network is clearly unreliable (procedure returns 0) or it is absolutely reliable (returns 1). In some cases it is possible to improve factoring process by calculating reliabilities of intermediate networks directly, i.e. without further factorization. For example, the formula 5-vertex graph reliability can be applied for $R(G)$ calculation [14].

For DCNR calculation we have a modified factoring method which is much faster than the basic factoring method (2) in the diameter constrained case [9]. The main feature of this method is operating with the list of paths instead of operating with graphs: in the preliminary step for any pair of terminals s and t the list $P_{st}(d)$ of all paths with limited length between s, t is generated. It automatically removes all the edges which don't belong to any such path from consideration. For example, all so called "attached trees" without terminals are no longer considered.

Afterwards all the operations described above are performed with the list of all paths $P = \cup_{s,t \in T} P_{st}(d)$. In these terms the success of the factored element will also make it absolutely reliable while failure of the factored element will remove all the paths which contained it from further consideration. Further on we refer to described method as CPFM (Cancela&Petingi factoring method).

Another approach for network reliability calculation is applying methods of reduction and decomposition. One of the most effective among them is the parallel-series transformation which removes chains and multiple edges from graph. For probabilistic connectivity [18,19] it is possible to apply such reduction in every call of the factoring procedure. For computing DCNR we can only use parallel-series transformation in the preliminary step, before factoring process starts [17]. Below we assume that the described reduction is performed during network reliability calculation by factoring method, both for probabilistic connectivity and DCNR.

4 Cumulative Updating of Network Reliability

Recent research [12] considered problem of determination whether a network is reliable enough in terms of network probabilistic connectivity (without diameter constraint). The idea of the proposed method is to check if a network is feasible without exact calculating a value of network reliability. For this purpose we define so called threshold R_0 which is a requirement of the network reliability. By RL and RU we will denote the lower bound and the upper bound of $R(G)$ respectively, and initialize them by 0 and 1. These bounds are updated in such a way that on i-th iteration $RL_i \geq RL_{i-1}$ and $RU_i \leq RU_{i-1}$. Decision process stops when either RL_l exceeds R_0 or R_0 exceeds RU_l. In the first case the network is supposed to be reliable and in the second one the network is unreliable.

Let us assume that during factoring procedure we obtain L final graphs $G_1, G_2, \ldots, G_L$, for which the reliability can be easily calculated. Let P_l for $1 \leq l \leq L$ be the probability to have G_l. Thus, $\sum_{l=1}^{L} P_l = 1$ and the following inequality holds for any $1 \leq k \leq L$ [12]:

$$\sum_{l=1}^{k} P_l R(G_l) \leq R(G) \leq 1 - \sum_{l=1}^{k} P_l (1 - R(G_l)). \tag{3}$$

This inequality gives the algorithm for cumulative updating of the lower and upper bounds of $R(G)$. Every time whenever reliability of some G_l for any $1 \leq l \leq L$ is calculated, we can update RL_l and RU_l in the following way:

$$\begin{aligned} RL_l &= RL_{l-1} + P_l R(G_l) \\ RU_l &= RU_{l-1} - P_l (1 - R(G_l)). \end{aligned} \tag{4}$$

RL_l and RU_l approach exact $G(R)$ value every time when l increases. Once either RL_l or RU_l reaches R_0, the proposed algorithm concludes the feasibility of G: if RL_l reaches R_0, G is feasible; if RU_l passes R_0, G is infeasible. Thus, we can set any acceptable value of R_0 in order to stop the method during execution without performing exact calculating of the network reliability.

We have applied this approach for DCNR bounds updating by CPFM. As it was mentioned above, CPFM doesn't oprate with graphs directly, instead it operates with the list of all paths P. In cases when either at least one pair of terminals cannot be connected by any path or all pairs of terminals are connected by absolutely reliable paths we can update our RL_l and RU_l values. In other words, these cases will play a role of the final graphs $G_1, G_2, \ldots, G_L$. We also denote by P_l the probability of the network obtained on l-th iteration. P_0 will be initialized by 1. Any time during the factoring procedure we should multiply P_l by either r_e or $1 - r_e$ depending on the factored element e status.

Parameters of the modified factoring procedure in CPFM aren't graphs. Instead we use 6 parameters, which describe the corresponding graph from the viewpoint of P_d. Listed below is the parameters of the CPFM and the pseudocode of the proposed method for DCNR bounds cumulative updating.

- np_{st}: the number of paths of length at most d between s and t in the graph being considered.
- $links_p$: the number of non-perfect edges (edges e such that $r(e) < 1$) in path p, for every $p \in P_d$.
- $feasible_p$: this is a flag, which has value False when the path is no longer feasible, i.e. it includes an edge which failed; and True otherwise.
- $connected_{st}$: this is a flag, which has value True when s and t are connected by a perfect path of length at most d and False otherwise.
- $connectedPairs$: this is the number of connected pairs of terminals (those between which there is a perfect path of length at most d).

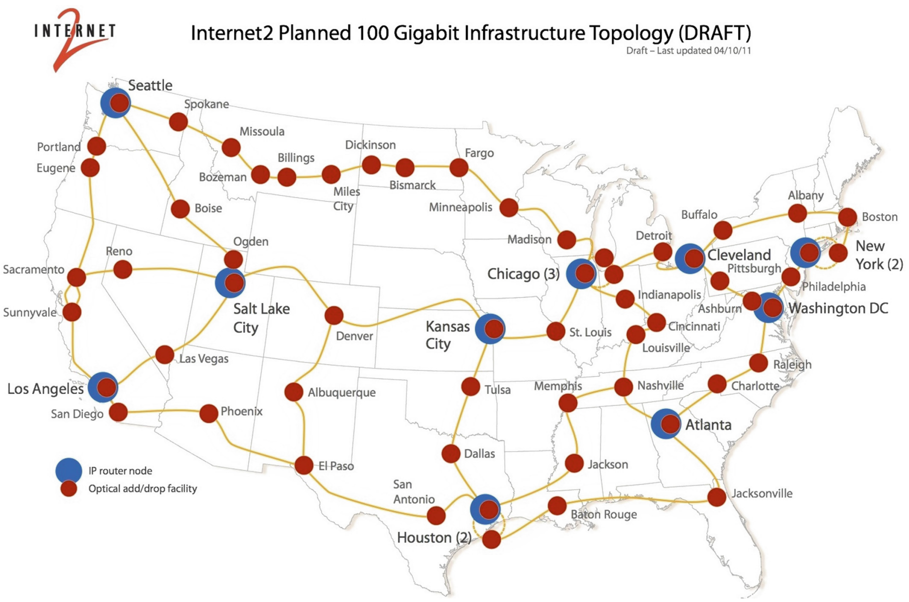

Fig. 1. Tested network (Color figure online)

```
Input: G = (V, E), d, P_d, P(e), np(s,t), links(p), feasible(p),
       connected(s,t), connectedPairs, RL = 0, RU = 1, P_l = 1
Function FACTO( np(s,t), links(p), feasible(p), connected(s,t),
connectedPairs, P_l)
    if nowTime − startTime > T_0 or RL > R_0 or RU < R_0 then
        return
    end
    e ← arbitrary edge : 0 < r_e < 1
    contractEdge( np(s,t), links(p), feasible(p), connected(s,t),
    connectedPairs, P_l)
    deleteEdge( np(s,t), links(p), feasible(p), connected(s,t),
    connectedPairs, P_l)
end
Function contractEdge( np(s,t), links(p), feasible(p), connected(s,t),
connectedPairs, P_l)
    P_l ← P_l * r_e
    foreach p = (s, ..., t) in P(e) such that feasible(p) = true do
        links(p) ← links(p) − 1
        if connected(s,t) = false and links(p) = 0 then
            connected(s,t) ← true
            connectedPairs ← connectedPairs + 1
            if connectedPairs = k×(k−1)/2 then
                RL ← RL + P_l
                return
            end
        end
    end
    FACTO (np(s,t), links(p), feasible(p), connected(s,t),
    connectedPairs, P_l)
end
Function deleteEdge( np(s,t), links(p), feasible(p), connected(s,t),
connectedPairs, P_l)
    P_l ← P_l * r_e
    foreach p = (s, ..., t) in P(e) such that feasible(p) = true do
        feasible(p) ← false
        np(s,t) ← np(s,t) − 1
        if np(s,t) = 0 then
            RU ← RU − P_l
            return
        end
    end
    FACTO (np(s,t), links(p), feasible(p), connected(s,t),
    connectedPairs, P_l)
end
```

1. Pseudocode of the method for cumulative updating of DCNR bounds

Diagram (Fig. 2) shows updating of RL and RU during execution of the proposed procedure for topology of Internet2 network (Fig. 1) for the diameter value 25. Edge reliability is equal 0.9 for each edge. R_0 value was equal to exact value of DCNR. Calculation time was about 48 s.

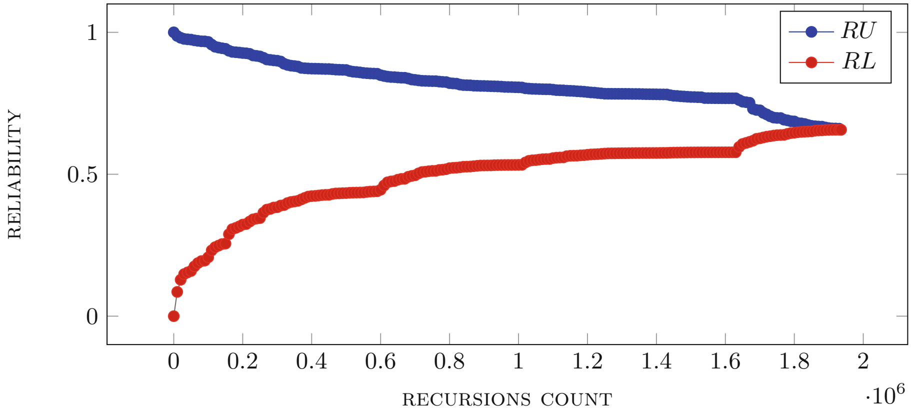

Fig. 2. D = 25, P = 0.9 (Color figure online)

5 Genetic Algorithm

Genetic Algorithms are a common probabilistic optimization method based on the model of natural evolution. Imitation of natural processes — selection, mutation, recombination, reproduction, proliferation is used as a basis, refer to Charles Darvin's theory presented in "On the Origin of Species" [20].

Algorithmic scheme (Fig. 3)**:** At first, individual is presented as chromosomes — sequenced collection of elements (more often as a bit string). Each chromosome presents some solution. Then, the population is defined like an arbitrary subset of full set of chromosomes. The most appropriate individuals are found by a fitness function. Next step is selection of chromosomes — choosing mates (for example, wheel selection). Crossover occurs among the fittest individuals specially selected for it. After crossover, new offspring goes through mutation. New population is combined from fittest individuals from the current population and new individuals with possible addition of randomly chosen new individuals. Algorithms stops after given time constraints, given number of generations, or when given number of generations does not produce any improvement of solution.

Cumulative updating lets the GA operators work faster due to cutting down bad chromosomes (with worse fitness function value).

Mutation: let A_0 be original chromosome with known diameter constrained reliability R_0. Verification new chromosome A_1:

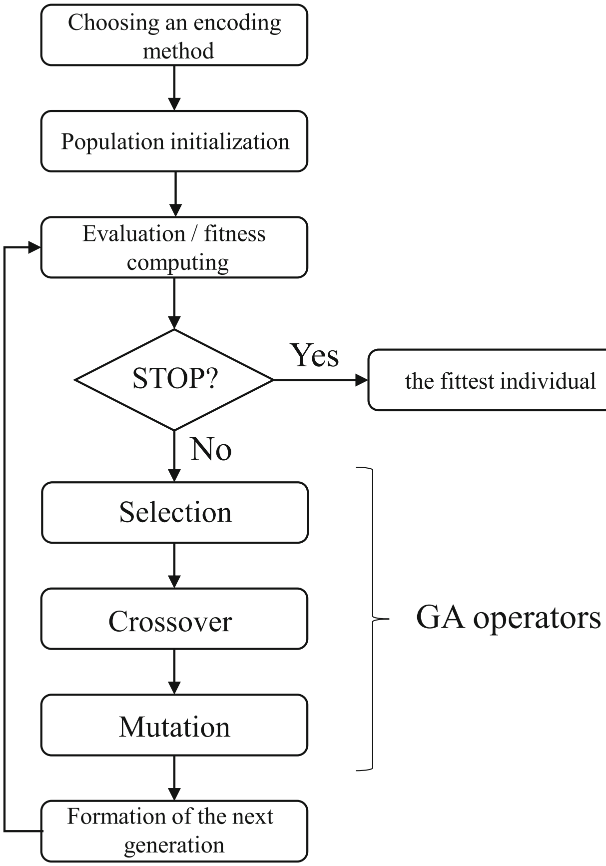

Fig. 3. Genetic algorithm scheme

$$Feas(A_1, R_0) = \begin{cases} 1, R_k^d(A_1) > R_0; \\ 0, else. \end{cases} \tag{5}$$

Crossover: let A_0 and A_1 be parents with reliability minimum of R_{min}. Verification offspring A_2:

$$Feas(A_2, R_{min}) = \begin{cases} 1, R_k^d(A_2) > R_{min}; \\ 0, else. \end{cases} \tag{6}$$

Obtained offspring is accepted only if it is better fitted than one of the parents.

6 Artificial Immune System

Artificial immune system (AIS) theory comes from theoretical immunology in the middle of 80th. First immune algorithms were used by Bersini [2] for solving different problems. The main aim of AIS is to use immunology principles for creating systems to solve different optimization problems [3].

One of the main algorithm among AIS algorithms is the Clonal Selection Algorithm (CSA) [5]. The CSA is simulating the natural B-cell response mechanism. When the antigen (virus, for instance) get to a blood, B-cells start to secret antibodies. Each cell secrets only one type of antibody specific for antigen. During the process the B-cells are cloning and mutating for achieving the best match for antigen. Those B-cells with better matching start quickly spread antibodies and become plasma cells, part of them becomes memory cells and is circulating in blood until new invasion.

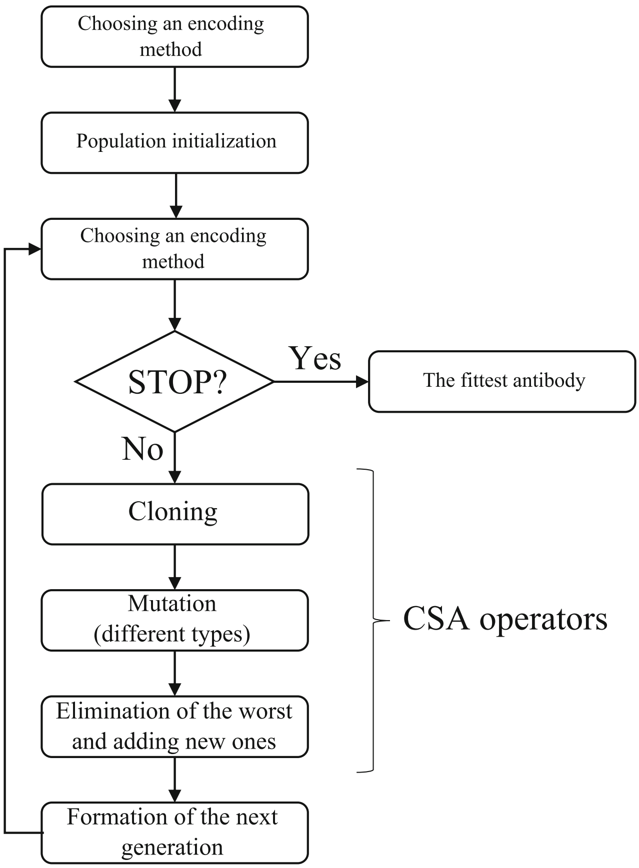

Fig. 4. Clonal selection algorithm scheme

Algorithmic scheme (Fig. 4)**:** Like chromosomes in GA, antibodies are coded as bit strings or as collection of elements. An arbitrary set of them is defined as the population. Then we calculate affinity of each antibody in the population. The next step is to clone antibodies accordingly their affinity values. Every clone goes through mutation to obtain better solution. After elimination of some part of the worst clones, new group of new random antibodies adds to population in the same number. Next, algorithm works repeatedly with new population until stop criteria: time constraints, number of generations etc.

Same as GAs we can accelerate clonal selection operators by cutting down worst antibodies using the cumulative updating.

7 Case Studies

This section presents series of experiments aimed to demonstrate relation between given data and running time. We demonstrate this on not dense graphs with 15 vertices (cause of their fast computational time). The results of the GA and CSA performance are presented in Tables 1 and 2 respectively. Changeable parameters are diameter (5, 8, 10) and edge reliability (0.1, 0.5, 0.9), which is the same for all edges. Other parameters fixed: mutation probability for GA is 0.1, size of population is 50, a number of populations is 10. So we have to calculate exactly reliabilities for 500 graphs for mutation and crossover. The number of terminals is equal to 3.

Table 1. Computational results for GA

Graph $\lvert V\rvert = 15$, $\lvert E\rvert = 22$		d = 5	d = 8	d = 10
r = 0,1	GA	1 m 55 s 789 ms	2 m 43 s 671 ms	3 m 50 s 336 ms
	GAwithCU	1 m 56 s 70 ms	2 m 43 s 563 ms	3 m 50 s 374 ms
	Reliability	0,053377031	0,035480363	0,042657732
r = 0,5	GA	39 s 247 ms	2 m 13 s 754 ms	12 m 18 s 777 ms
	GAwithCU	39 s 321 ms	2 m 11 s 322 ms	12 m 21 s 311 ms
	Reliability	0,801047325	0,796380579	0,830094337
r = 0,9	GA	1 m 59 s 10 ms	3 m 5 s 699 ms	4 m 1 s 559 ms
	GAwithCU	1 m 54 s 263 ms	2 m 41 s 897 ms	3 m 27 s 677 ms
	Reliability	0,999966901	0,999951435	0,999883982

We've also compared computational results in case of different numbers of terminals (3 and 7). The results are presented in Table 3.

We can see that using GAs and CSAs with cumulative updating for such types of problems gives some improvements, especially in the case of not dense and highly reliable graphs. The best results were obtained for problems where amount of terminals is large.

Table 2. Computational results for CSA

Graph $\|V\| = 15$, $\|E\| = 22$		d = 5	d = 8	d = 10
r = 0,1	CSA	42 s 939 ms	1 m 18 s 686 ms	1 m 20 s 586 ms
	CASwithCU	43 s 484 ms	1 m 19 s 501 ms	1 m 20 s 227 ms
	Reliability	0,033715333	0,035687813	0,035632029
r = 0,5	CSA	46 s 870 ms	1 m 46 s 51 ms	2 m 14 s 788 ms
	CSAwithCU	46 s 965 ms	1 m 44 s 935 ms	2 m 5 s 659 ms
	Reliability	0,771484375	0,756820679	0,652519226
r = 0,9	CSA	44 s 178 ms	1 m 52 s 241 ms	2 m 50 s 238 ms
	CSAwithCU	43 s 616 ms	1 m 45 s 308 ms	2 m 31 s 871 ms
	Reliability	0,999604889	0,999946036	0,999672344

Table 3. Computational results for $T = 3$ and $T = 7$

	$\|T\| = 3$	$\|T\| = 7$
GA	3 m 5 s 699 ms	39 m 20 s 844 ms
GAwithCU	2 m 41 s 897 ms	33 m 29 s 853 ms
Reliability	0,999951434	0,998523183
CSA	1 m 52 s 241 ms	13 m 34 s 306 ms
CSAwithCU	1 m 45 s 308 ms	9 m 53 s 614 ms
Reliability	0,999946036	0,992177548

8 Conclusion

Proposed optimization approach allows to reduce computational time for obtaining an appropriate solution, i.e. a reliable enough network topology. Nevertheless, network topology optimization problems still show a great level of complexity. Our ongoing research involves studying of new improvements of cumulative updating method for further speeding up the optimization process.

References

1. Holland, J.H.: Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann. Arbor (1975)
2. Bersini, C., Varela, F.J.: Hints for adaptive problem solving gleaned from immune networks. In: Schwefel, H.-P., Männer, R. (eds.) PPSN 1990. LNCS, vol. 496, pp. 343–354. Springer, Heidelberg (1991)
3. Ishida, Y.: The immune system as a self-identification process: a survey and a proposal. In: Proceedings of the ICMAS International Workshop on Immunity-Based Systems, pp. 2–12 (1996)

4. Landwehr, B.: A genetic algorithm based approach for multi-objective data-flow graph optimization. In: Asia and South Pacific Design Automation Conference 1999 (ASP-DAC 1999), vol. 1, pp. 355–358 (1999)
5. De Castro, L.N., Von Zuben, F.J.: The clonal selection algorithm with engineering applications. In: Workshop Proceedings of GECCO. Workshop on Artificial Immune Systems and Their Applications, Las Vegas, USA, pp. 36–37 (2000)
6. Kumar, A., Mishra, K.K., Misra, A.K.: Optimizing the reliability of communication network using specially designed genetic algorithm. In: World Congress on Nature & Biologically Inspired Computing, NaBIC 2009, pp. 499–502 (2009)
7. Ball, M.O.: Computational complexity of network reliability analysis: an overview. IEEE Trans. Reliab. **35**, 230–239 (1986)
8. Petingi, L., Rodriguez, J.: Reliability of networks with delay constraints. Congressus Numerantium. **152**, 117–123 (2001)
9. Cancela, H., Petingi, L.: Diameter constrained network reliability: exact evaluation by factorization and bounds. In: International Conference on Industrial Logistics, Okinawa, Japan, pp. 359–356 (2001)
10. Pandurangan, G., Raghavan, P., Upfal, E.: Building low-diameter peer-to-peer networks. IEEE J. Sel. Areas Commun. (JSAC) **21**(6), 995–1002 (2003)
11. Canale, E., Cancela, H., Robledo, F., Romero, P., Sartor, P.: Full complexityanalysis of the diameter-constrained reliability. Int. Trans. Oper. Res. **22**(5), 811–821 (2015)
12. Won, J.-M., Karray, F.: Cumulative update of all-terminal reliability for faster feasibility decision. IEEE Trans. Reliab. **59**(3), 551–562 (2010)
13. Won, J.-M., Karray, F.: A greedy algorithm for faster feasibility evaluation of all-terminal-reliable networks. IEEE Trans. Syst. Man Cybern. Part B Cybern. **41**(6), 1600–1611 (2011)
14. Rodionov, A.S., Migov, D.A., Rodionova, O.K.: Improvements in the efficiency of cumulative updating of all-terminal network reliability. IEEE Trans. Reliab. **61**(2), 460–465 (2012)
15. Nechunaeva, K.A., Migov, D.A.: Speeding up of genetic algorithm for network topology optimization with use of cumulative updating of network reliability. In: ACM IMCOM2015, article 42. ACM New York (2015)
16. Migov, D.A., Rodionov, A.S.: Parallel implementation of the factoring method for network reliability calculation. In: Murgante, B., et al. (eds.) ICCSA 2014, Part VI. LNCS, vol. 8584, pp. 654–664. Springer, Heidelberg (2014)
17. Migov, D.A., Nesterov, S.N.: Methods of speeding up of diameter constrained network reliability calculation. In: Gervasi, O., Murgante, B., Misra, S., Gavrilova, M.L., Rocha, A.M.A.C., Torre, C., Taniar, D., Apduhan, B.O. (eds.) ICCSA 2015, Part II. LNCS, vol. 9156, pp. 121–133. Springer, Heidelberg (2015)
18. Shooman, A.M., Kershenbaum, A.: Exact graph-reduction algorithms for network reliability analysis. In: IEEE Global Telecommunications Conference GLOBECOM 1991, pp. 1412–1420. IEEEP Press, New York (1991)
19. Rodionova, O.K., Rodionov, A.S., Choo, H.: Network probabilistic connectivity: exact calculation with use of chains. In: Laganá, A., Gavrilova, M.L., Kumar, V., Mun, Y., Tan, C.J.K., Gervasi, O. (eds.) ICCSA 2004. LNCS, vol. 3045, pp. 315–324. Springer, Heidelberg (2004)
20. Darwin, C.R.: On the Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life. John Murray, London (1859)