International Journal of Computational Intelligence and Applications

Reliable Communication Network Design with Evolutionary Algorithms

Dirk Reichelt

Institute of Information Systems, Ilmenau Technical University
98684 Ilmenau, Germany,
Dirk.Reichelt@tu-ilmenau.de,

Franz Rothlauf

Department of Information Systems I, University of Mannheim
68131 Mannheim, Germany
rothlauf@uni-mannheim.de

Received (received date)
Revised (revised date)

For the reliable communication network design (RCND) problem links are unreliable and for each link several options are available with different reliabilities and costs. The goal is to find a cost-minimal communication network design that satisfies a pre-defined overall reliability constraint. This paper presents two new EA approaches, LaBORNet and BaBORNet, for the RCND problem. LaBORNet uses an encoding that represents the network topology as well as the used link options and repairs infeasible solutions using an additional repair heuristic (CURE). BaBORNet encodes only the network topology and determines the link options by using the repair heuristic CURE as a local search method. The experimental results show that the new EA approaches using repair heuristics outperform existing EA approaches from the literature using penalties for infeasible solutions and find better solutions for existing problems from the literature as well as for new and larger test problems.

Keywords: evolutionary algorithms; network design; repair heuristics, all-terminal reliability.

1. Introduction

The design of communication networks is a complex optimization problem for telecommunication companies and has strong impact on their economic success. For the construction of communication networks different types of communication links are available which typically differ in their costs and their reliability. The reliability of a link measures the probability that it is available and can be used for the transport of data. In practice, all communication links are vulnerable to failures and the cost of a link is increasing with higher reliability (and lower failure probability). Network designers are confronted with the problem to construct communication networks such that the cost of the resulting network is minimized and the overall reliability is above some pre-defined threshold. A common measurement for the overall reliability of a communication network is the *all-terminal reliability*. It is defined as the probability that all nodes in the network keep connected, given the probability of success/failure for each node and link in the network.[5] The resulting network design problem is to find a communication network that minimizes the overall network costs under a given reliability constraint. Both, the reliable communication network design (RCND) problem as well as the calculation of the all-terminal reliability has been proven as NP-hard.[12,24] In the past, heuristic optimization methods and especially evolutionary algorithms (EA) have already been applied to the RCND problem.[7,4,16,8,21]

This paper presents two new EA-based approaches for the RCND problem and compares their performance to existing approaches. Both new approaches (LaBORNet and BaBORNet) use repair strategies that ensure that the all-terminal reliability of the resulting network design is above some pre-defined threshold. The two new approaches differ in the handling of the all-terminal reliability constraint. In LaBORNet an EA determines the network topology as well as the types of communication links that are used. In BaBORNet the EA only determines the topology and a local search strategy assigns the types of communication links. A comparison to existing EA approaches that use penalties for infeasible solutions shows for existing and new test problem instances that both new approaches show significantly higher performance. A comparison between LaBORNet and BaBORNet shows that for small RCND problems LaBORNet is faster and finds better solutions. However, with increasing problem size, combining an EA that determines the topology with a local search strategy that determines the types of link (BaBORNet) is more efficient as the search space of the EA is smaller.

The following section defines the RCND problem and reviews measurements for network reliability and existing approaches for the RCND problem. Section 3 describes a repair heuristic (local search) that ensures the feasibility of solutions and presents the two new EA approaches (LaBORNet and BaBORNet). Experimental results and a comparison to existing approaches for the RCND problem are presented in section 4. In section 5 the paper closes with a short conclusion.

2. Design of Cost-minimal Communication Networks under Reliability Constraints

2.1. *The Reliable Communication Network Design Problem*

The reliable communication network design (RCND) problem seeks a network design with minimal costs under a given reliability constraint. The network design consists of the network topology and the type of links that are used for the edges. The topology of a network N is modeled as an undirected graph $G(V, E)$ with V is the set of vertices and E is the set of possible edges. n is the number $|V|$ of nodes. The position of the nodes is fixed and node setup costs are not considered. For each edge $e_{ij} \in E$ between node i and j several link options l_k $(k = 1, \dots, k_{max})$ with different reliabilities $r(l_k(e_{ij}))$ and costs $c(l_k(e_{ij}))$ are possible. $l_k(e_{ij})$ is the link option k chosen for edge e_{ij}. It is assumed that all nodes are perfect reliable, an edge can be either in the state $s_{e_{ij}}$ "operational" ($s_{e_{ij}} = 1$) or "failed" ($s_{e_{ij}} = 0$), link failures are independent, and all links are bidirectional. A candidate solution for the RCND problem is represented by a subgraph $G_N(V, E_N \subset E)$ and the set of link options l_k that are used for the $e_{ij} \in E_N$. The objective function is:

$$C(G_N) = \sum_{e_{ij} \in E_N} c(l_k(e_{ij})) \rightarrow min, \quad \text{with: } R(G_N) \geq R_0 \text{ (reliability constraint)}, \quad (1)$$

where $C(G_N)$ is the total cost of a network design summarizing the costs $c(l_k(e_{ij}))$ of all links $e_{ij} \in E_N$. $R(G_N)$ is the overall reliability of the network and R_0 is the minimal required reliability.

2.2. *Reliability Measurements for Communication Networks*

We give a short overview about connectivity and other reliability measurements for communication networks that are used in the paper. As mentioned before we assume that nodes are perfectly reliable and only edges can fail.

2.2.1. *Connectivity*

The *connectivity*[6] of a network describes if a fully connected network is still fully connected if one or more edges fail. A connected network component is a subset of nodes that remains connected after the failure of one or more edges. A network is edge 1-connected if there is at least one distinct path between each node pair in the network. A failure of one edge already disconnects an edge 1-connected network. Fig. 1 shows an example with 6 nodes. The network is edge 1-connected as the failure of edge $e_{1,4}$ separates the network in two unconnected components {1,2,3} and {4,5,6}. The connectivity of the example network can be increased by adding one more edge (e.g. $e_{2,5}$). Then, the network is edge 2-connected as there are two edge-disjoint paths between any node pair.

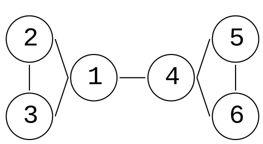


Fig. 1. Example network

In general, in an edge n-connected network there are at least n edge-disjoint paths for each node pair and it remains connected if less than n edges fail. Using the connectivity of a network as a measurement for its reliability allows only to consider the survivability of a network and no failure or reliability probabilities can be given. Therefore, probability-based reliability measurements have been developed that consider the failure probabilities of the links.

2.2.2. *All-Terminal Reliability*

The limitations of the simple connectivity measurements are overcome by the source-terminal reliability and k-terminal reliability.[6] These reliability measurements describe the probability that two or k nodes of a network are connected. The all-terminal reliability R_{All} [6] is equivalent to the n-terminal reliability and is the probability that all n nodes of a communication network are connected ($n = |V|$). Therefore, R_{All} is the probability that there is a path between each node pair in G_N.[5] The all-terminal reliability R_{All} is an appropriate measurement for the reliability of a network design based on imperfect edges.[6]

The exact calculation of R_{All} is NP-hard.[24] In the literature a variety of exact and approximative methods for calculating R_{All} have been proposed.[19,11,15,5,17] In this work different methods depending on the size of the network problem are used. To check if the reliability constraint is fulfilled ($R(G_N) \geq R_0$) the upper bound from Konak and Smith[15] is used. If the upper bound is larger than R_0, $R(G_N)$ is calculated using the exact decomposition approach proposed by Chen[5] if the problem has less than 15 nodes. Due to the large computational effort of this approach it is substituted by a Monte-Carlo simulation technique from Fishman[11] if $n \geq 15$.

2.3. *Existing Solution Approaches for the RCND problem*

In the literature, several approaches using heuristic optimization methods have been proposed for the RCND and slightly modified problems. Dengiz et al.[9] proposed a genetic algorithm (GA) for a simplified RCND problem with only one possible link option l_k ($k_{max} = 1$) using a penalty function for infeasible solutions G_N, where $R(G_N) < R_0$. This approach was extended by Baran and Laufer[3] who parallelized the GA to solve larger problem instances. Reichelt et al.[21] examined both approaches and found that for several problem instances only infeasible networks are found as best solutions. The penalty for infeasible networks designs is too low and infeasible solutions can have higher fitness than the optimal (feasible) solution. To overcome the problems of the penalty approach, Reichelt et al. proposed a GA using a repair function. Dengiz and Alabap[8] presented a simulated annealing approach for the RCND problem which showed better performance in comparison to the previous GA approaches.

The approaches presented by Dengiz et al. are limited to only one possible link type. This means the network designer can not select different link types for different edges of the network but only decide if there is a link between node i and j or not (0/1 problem). Therefore, a solution for the RCND problem is determined by the network topology alone and no decisions about the used link type are possible. However in reality, network designers can usually choose from several link types with different reliabilities and costs. In general, links with low reliability are cheap and links with high-reliability are extremely expensive.

Baran et al.[10,2] addressed several possible link types and solved the RCND problem as a multi-objective problem using a multi-objective GA. In this approach there are several link types available and the objectives, overall cost of the network and reliability, are optimized in parallel. As the computational effort of this multi-objective approach is high, Baran et al. used a Monte-Carlo simulation for the evaluation of

R_{All} using only 10,000 replications. Due to the small number of replications the variance of the estimation $\hat{R}_{All}(G_N)$ of the all-terminal reliability $R_{All}(G_N)$ is high and the results for high-reliable communication networks ($R_{All}(G_N) \approx 0.99$) are inaccurate. The RCND problem with a reliability constraint ($R_{All}(G_N) \geq R_0$) and different link options has also been investigated by Deeter and Smith.[7] Similarly to Dengiz et al. the authors proposed a GA using a penalty function. The penalty has to be adjusted for each network problem using a penalty rate parameter. Deeter and Smith recommend that the penalty should be set according to the problem complexity but give no advice on how the problem complexity can be measured or how the parameter can be chosen for a specific problem. Therefore, using this method requires either additional knowledge about the problem complexity or additional experiments for finding appropriate penalty values.

3. Evolutionary Algorithms using Repair Concepts for the Design of Reliable Communication Networks

When using heuristic optimization methods like evolutionary algorithms for the RCND problem there are different strategies[18] for handling infeasible networks that violate the reliability constraint (compare equation 1). Infeasible solutions can be either removed from the population, repaired such that they become feasible, or the fitness of infeasible solutions can be reduced by additional penalties. To just remove infeasible solutions from the population is not appropriate as most randomly created networks are infeasible and the optimal solution is at the boundary between feasible and infeasible solutions. Therefore, previous approaches mostly used penalties for infeasible solutions. However, the analysis of existing penalty approaches for the RCND problem[21] as well as penalty approaches for other constraint optimization problems[13] shows that the proper design of penalties is difficult and EAs are often misled. Therefore, we propose to use repair approaches for the RCND problem. Consequently, repair mechanisms are developed that are applied to infeasible solutions and output a valid network.

3.1. *CURE - A Deterministic Cut-Based Repair Heuristic*

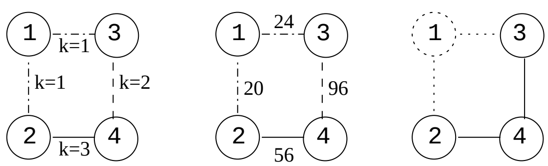
The purpose of the cut-based repair heuristic (CURE) is to repair infeasible solutions G_N of the RCND problem that violate the reliability constraint ($R_{All}(G_N) < R_0$). CURE performs iterative repair steps and increases in each step the overall reliability of the network until $R_{All}(G_N) \geq R_0$. To increase the overall reliability R_{All}, CURE iteratively increases the reliability of the edges e_{ij} by choosing a link option l_k with a higher reliability $r(l_k(e_{ij}))$. If the reliability of all edges $e_{ij} \in G_N$ is maximal and still $R_{All}(G_N) < R_0$, CURE iteratively adds additional edges to G_N using the procedure proposed by Reichelt et al.[21]

The concept of CURE is based on minimal cuts in graphs. For a graph $G_N(V, E_N)$ a cut $C \subset V$ is defined as a non-empty subset of the nodes V. Each cut

C determines a set of edges $E_C \subset E$, with $\forall e_{ij} \in E_C : i \in C$ and $j \notin C$. Removing all edges E_C from G_N would split the graph in two subgraphs with the vertex sets C and $V \setminus C$. Therefore, the failure of all edges $e_{ij} \in E_C$ would disconnect the network. The weight of a cut C is the sum of weights of the edges $e_{ij} \in E_C$. A minimal cut (MinCut) is the cut with minimal weight. In CURE, minimal cuts are used to find a set of edges, where an increase in reliability (choosing more reliable link types) can be achieved with minimal cost. Therefore, the weight of an edge $e_{ij} \in G_N$ is the cost $c(l_{k+1}(e_{ij}))$, where k is the currently used link option and $k+1$ is the next more reliable link option (the reliabilities of the links increase monotonously with larger k). As a result, the edges contained in the minimal cut represent a set of edges that increase the all-terminal reliability of the network with minimal costs. CURE process the following steps:

Input: $G_N(V, E_N)$, $G(V, E)$, R_0
Queue $Q = \emptyset$
Q.append(G_N)
while (!Q.empty) & ($R_{All}(G_N) < R_0$)) do begin
  G_{work} = Q.first()
  assign weights (costs) to $e_{ij} \in G_{work}$: $c(l_k(e_{ij})) = \begin{cases} c(l_{k+1}(e_{ij})) & \text{if } k < k_{max} \\ c(l_k(e_{ij})) & \text{if } k = k_{max} \end{cases}$
  C = MinCut(G_{work}) (using the weights $c(l_k(e_{ij}))$
  increase reliability $\forall e_{ij} \in E_C : l_k(e_{ij}) = \begin{cases} l_{k+1}(e_{ij}) & \text{if } k < k_{max} \\ l_k(e_{ij}) & \text{if } k = k_{max} \end{cases}$
  calculate $R_{All}(G_N)$
  $G_{N_1} = G_{work} \setminus \{C\}, G_{N_2} = C$
  if number of nodes in $(G_{N_1}) > 1$
    Q.append(G_{N_1})
  if number of nodes in $(G_{N_2}) > 1$
    Q.append(G_{N_2})
  Q.remove(G_{work})
  if (Q.empty) & ($\exists e_{ij} \in E_N$: $k < k_{max}$)
    Q.append(G_N)
end
if ($R_{All}(G_N) < R_0$) begin
  add $e_{ij} \in E \setminus E_N$ to G_N
  $\forall e_{ij} \in G_N : l_k(e_{ij}) = l_1(e_{ij})$
  call CURE
end

In a first step, CURE assigns weights to all edges $e_{ij} \in G_N$. The weights of the edges are the cost $c(l_{k+1}(e_{ij}))$ of the next more reliable link option (k is the number of the current option). If the reliability of an edge is already maximal ($k = k_{max}$), the cost of the currently chosen link option is used. Then, CURE calculates[22] the minimal cut C using the weights $c(l_k(e_{ij}))$. As a result, we get a set of nodes C and a set of edges E_C. In a next step, for all edges $e_{ij} \in E_C$ the next more reliable link options are chosen. If the use of more reliable link options allows G_N to fulfill the reliability constraint, CURE terminates. Otherwise, CURE considers the two subgraphs G_{N_1} and G_{N_2}, that are created by the removal of all cut edges E_C from G_N. Both subgraphs are added to the queue if they have more than one node. If the queue is empty, $R_{All}(G_N) < R_0$, and there are still some e_{ij} whose reliability can be increased, G_N is added again to the queue. If for all edges in G_N the link options with the maximal reliability are chosen and still $R_{All}(G_N) < R_0$ an additional heuristic[21] adds more edges to G_N and CURE is called again.



	$l_1(e_{ij})$		$l_2(e_{ij})$		$l_3(e_{ij})$	
	r	c	r	c	r	c
$e_{1,2}$	0.8	10	0.9	20	0.95	40
$e_{2,4}$	0.8	14	0.9	28	0.95	56
$e_{3,4}$	0.8	24	0.9	48	0.95	96
$e_{1,3}$	0.8	12	0.9	24	0.95	48

(a) G_N with link options k

(b) G_N with weights $c(l_k(e_{ij}))$

(c) subgraph G_{N_1}

(d) reliability $r(l_k(e_{ij}))$ and cost $c(l_k(e_{ij}))$ for different link options l_k

Fig. 2. Functionality of the CURE repair heuristic

Fig. 2 illustrates the functionality of CURE. There are three different link options ($k_{max} = 3$). Fig. 2(d) lists the reliabilities $r(l_k(e_{ij}))$ and costs $c(l_k(e_{ij}))$ of the different options. The topology and the used link options k for an example network G_N are shown in Fig. 2(a). Using the numbers from Fig. 2(d) CURE assigns weights to all e_{ij} (compare Fig. 2(b)). Based on the weights, the minimum cut finds the node set C such that the sum of the weights of the links $e_{ij} \in E_C$ is minimal. When removing the links $e_{ij} \in E_C$ from G_N the network becomes disconnected. The minimal cut is $C = \{1\}$ with $E_C = \{e_{1,2}, e_{1,3}\}$ and a weight of 44 (indicated by the dashed lines in Fig. 2(c)). In the next step the reliability of $e_{1,2}$ and $e_{1,3}$ is increased to $r(e_{1,2}) = 0.9$ and $r(e_{1,3}) = 0.9$ and $R_{All}(G_N)$ is calculated. If $R_{All}(G_N) < R_0$, CURE continues with the graph G_{N_1}. If the queue is empty and still $R_{All}(G_N) < R_0$, G_N is appended to the queue. This procedure is repeated until all edges $e_{ij} \in E_N$ have maximum reliability. Then, additional edges are added to G_N, all link options are set to the most unreliable value ($k = 1$), and CURE is called again.

3.2. *Using Repair Heuristics for Evolutionary Algorithms*

The purpose of a repair heuristic is to modify an infeasible solution for the RCND such that it becomes valid and fulfills the reliability constraint. There are two different possibilities for the repair process: either the repaired and valid solution completely replaces the infeasible solution[18] or the infeasible solution remains in the population and only the fitness of the repaired and valid solution is assigned to the infeasible solution,[20] which remains in the population. These two possibilities on how to perform the repair process follow the notion of Lamarckian evolution versus the Baldwin effect. In Lamarckian evolution[23] each individual bequeathes the learned improvements (repairs) to its offspring. That means that a repair strategy based on Lamarckian evolution repairs an infeasible solution and replaces it by the repaired solution. In contrast, the Baldwin effect[1] is based on the assumption that only the individual's fitness is changed by the learned improvements (repair) but the changes itself are not inherited to the offspring. Therefore, the underlying genotype remains untouched (the infeasible solution remains in the population) and only the fitness value of the valid solution is assigned to the infeasible individual.

In the following two sections, we present two EA approaches (LaBORNet and BaBORNet) for the RCND problem which use a repair mechanism (compare section 3.1) and are inspired by the Lamarckian evolution and the Baldwin effect respectively:

t:=0
create initial population P(t)
evaluate P(t), repair individuals with $R_{All} < R_0$
repeat until stop criteria
    t := t+1
    P*(t) := select individuals from P(t)
    P'(t) := recombine P*(t)
    P"(t) := mutate P'(t)
    evaluate P"(t), repair individuals with $R_{All} < R_0$
    P(t+1) := choose best individuals from P(t) and P"(t)

During the evaluation step each solution is checked if it fulfills the reliability constraint $R_{All}(G_N) \geq R_0$. If a solution violates the constraint it is repaired (compare section 3.1) such that the reliability constrained is fulfilled.

3.3. *LaBORNet - A Lamarckian Based Optimizer for Reliable Network Design Problems*

LaBORNet is inspired by the Lamarckian evolution and describes an EA where all infeasible solutions are replaced by repaired, valid, solutions. In LaBORNet each solution is encoded as a vector g with length $n(n-1)/2$, where $n = |V|$ is the number of nodes. Each element of g corresponds to a possible edge e_{ij} and indicates the number k of the link option $l_k(e_{ij})$ $(1 \leq k \leq k_{max})$ that is chosen for the edge e_{ij}. $g_i = 0$ indicates that there exists no link. Fig. 3 illustrates the encoding of a network. Dashed lines indicate edges that do not exist in the network ($g_i = 0$).

Randomly created solutions as well as solutions created by standard crossover (e.g. uniform or one-point crossover) or mutation operators (e.g. exchange or modification of alleles) can result in infeasible solutions. Such solutions are repaired by

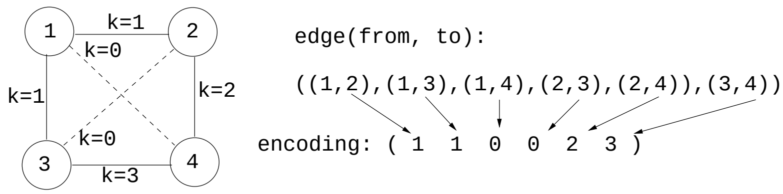


Fig. 3. LaBORNet encoding of a network

the approach presented in section 3.1 and replace the infeasible solution. LaBORNet is a combination of EA and local search where each EA individual encodes the topology of the network as well as the link options that are used for the links. The resulting search space is large, $(k_{max} + 1)^{n(n-1)/2}$.

3.4. *BaBORNet - A Baldwin Based Optimizer for Reliable Network Design Problems*

BaBORNet is inspired by the Baldwin effect and combines an EA with a local search (repair heuristic) that creates valid solutions. Individuals only encode the topology of the network and contain no information about the link types that are used for the edges. Therefore, the EA can only determine the topology of the network and the used link types are chosen by the repair strategy described in the next section. Fig. 4 shows the encoding of the same network as figure 3. The topology of a solution is encoded as a binary string g of length $n(n-1)/2$, where each allele corresponds to an edge $e_{ij} \in E$.

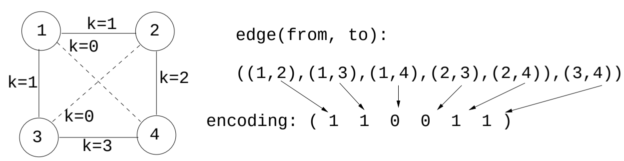


Fig. 4. BaBORNet encoding of a network

For the fitness evaluation of an individual each link type is set to the cheapest possible link option. If the resulting network violates the reliability constraint ($R_{All} < R_0$) the repair procedure from section 3.1 is applied. The repair heuristics changes the link options (chooses more reliable but more expensive link types) and adds additional edges to the network to obtain a valid solution. The repair heuristic outputs a network that satisfies the reliability constraint. Finally, the fitness of the repaired and valid solution is assigned to the individual. If the repair heuristics has added additional links to the network the bitstring representing the topology of the network also is changed and the additional links are added.

In comparison to the LaBORNet approach presented in the previous section the search space of the EA in BaBORNet is smaller ($2^{n(n-1)/2}$). For example, the search space of a small problem with 10 nodes and three link types is $2^{45} \approx 3.5 * 10^{13}$ for BaBORNet but $4^{45} \approx 1.2 * 10^{27}$ for LaBORNet. Due to the smaller size of the search space the EA is expected to find better network topologies. The BaBORNet approach can be seen as a combination of an EA which is responsible for finding good topologies and a local search (repair heuristic) which is responsible for finding proper link types. Despite the smaller EA search space, the computational effort of BaBORNet is higher as almost all solutions are initially infeasible and have to be repaired. As the repair process is time-consuming and has to be invoked for all fitness evaluations, the overall running time of BaBORNet is expected to exceed the overall running time of LaBORNet.

4. Experiments and Results

4.1. *Test Problems*

The performance of the EAs proposed in section 3 is evaluated for six test instances representing artificial and real world design problems:

Deeter10 (10 nodes): This test problem was created by Deeter and Smith.[7] The nodes are randomly placed on a 100x100 grid and there are three different link options with reliabilities $r(l_1) = 0.7$, $r(l_2) = 0.8$, and $r(l_3) = 0.9$ and corresponding costs $c(l_1) = 8$, $c(l_2) = 10$, and $c(l_3) = 14$ for each edge. The cost of a link with reliability $r(l_k)$ is calculated as the Euclidean distance between two nodes multiplied by $c(l_k)$. The best solution for $R_0 = 0.95$ published by Deeter and Smith[7] is 5,661.

Turkey19 (19 nodes): This problem represents a simplified version of a real-world design problem of the Turkish government.[7] The goal is to find a network with $R_0 = 0.99$ that interconnects 19 academic centers in 9 cities in Turkey. The cost of the best solution for this problem found by Deeter and Smith[7] was 7,694,708. This result was improved by Baran et al.[2] to 1,755,474 using a multi-objective GA.

ger15, ger20, ger25, ger30 (15, 20, 25, 30 nodes): We introduce four new test instances representing the 15, 20, 25 and 30 largest cities in Germany. There are three link options ($k_{max} = 3$) with different reliabilities ($r(l_1) = 0.7$, $r(l_2) = 0.8$, $r(l_3) = 0.9$) and corresponding costs ($c(l_1) = 8$, $c(l_2) = 10$, $c(l_3) = 14$). As for Deeter10, the cost of a link is the Euclidean distance between the nodes multiplied with $c(l_k)$. We present results for $R_0 = 0.95$.

4.2. *Experimental Design*

We performed experiments for the test problems described in the previous section and compared the performance of LaBORNet and BaBORNet with the penalty-based EA from Deeter and Smith.[7] For the experiments a steady state EA (with 50% overlapping populations), uniform crossover and allele-flipping mutation was used. The EA was stopped after 200 generations or if there was no fitness improvement over the last 20 generations. The crossover probability p_{cross} was 0.9 and the mutation probability p_{mut} was set to 0.01. The individuals in the initial population were randomly created using a propability of $P = 0.4$ for creating a link between two nodes. For LaBORNet and for the penalty-based EA all initial link option were chosen uniformly with probability $P = 1/k_{max}$. For each problem, 10 independent runs have been performed. For Deeter10, R_{All} was calculated using an exact decomposition method[5]; as the calculation of R_{All} is $\mathcal{NP}$-complete, for all other problems the exact method was replaced by a Monte-Carlo simulation.[11,17] The Monte-Carlo simulation starts with 30,000 samples which are increased by 1,000 each fifth generation to get more accurate predictions in later stages of the search.

Table 1 shows the experimental results for the different test problems. The tables show the EA type, the used population size *pop*, the cost $C(G_{best})$ of the best solution found at the end of a run (mean and standard deviation over 10 runs and best found solution), the average running time t_{conv} in seconds, and the number of fitness evaluations (average and standard deviation). For LaBORNet and BaBORNet we present results for *pop* = 100 and *pop* = 200. As a benchmark, we present results for the penalty EA for *pop* = 200. The cost of the best ever found solution is marked bold.

4.3. *Results*

A direct comparison to the best found results from the literature shows that by using BaBORNetand LaBORNet the cost of the best found solution can be reduced from 5,661[7] to 4,386 for the Deeter10 problem and from 1,755,474[2] to 1,624,960 for the Turkey19 problem. This is a significant improvement. Interestingly, the results obtained for our implementation of the penalty approach from Deeter and Smith[7] are better than the results reported in the original work. This can be explained by two modifications. Firstly, we create in the initial population a link between two nodes with probability 0.4 instead of 0.75. This modified initialization strategy makes use of the fact that high-quality solutions use only a low number of links. Secondly, the use of a steady state EA with overlapping populations instead of a standard generational EA results in higher selection pressure what leads to higher EA performance than reported by Deeter and Smith.

The results show that the use of repair approaches like BaBORNet or LaBORNet for the RCND problem allows to find better solutions than the use of the penalty approach from Deeter and Smith[7] or the multi-objective approach from Baran et al.[2] Furthermore, in all problem instances both repair approaches significantly outperform the penalty approach. The increase in solution quality for *pop* = 200 ranges from about 10% for small problems with 10 nodes (Deeter10) to 40% for problems with 30 nodes (ger30).

A direct comparison between LaBORNet and BaBORNet reveals that BaBORNet, that means combining an EA for finding good topologies with a local search strategy for determining appropriate link types, outperforms LaBORNet with increasing problem size n. Only for the small Deeter10 problem instance, LaBORNet outperforms BaBORNet. This can be explained by the increase of the search space with larger n. When using LaBORNet, the resulting search space is much larger as when using BaBORNet and the EA has to find both, a high-quality network

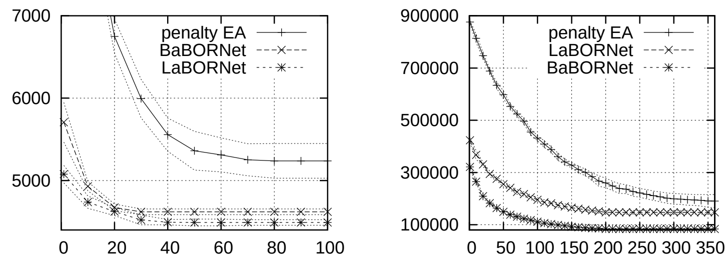
Table 1. Experimental results

problem	EA type	*pop*	cost $C(G_{best})$ at end of run mean (std.dev.)	cost $C(G_{best})$ at end of run best	t_{conv} (in sec)	evaluations mean (std.dev.)
Deeter10	LaBORNet	100	4,501 (70)	4,433	710	2,640 (512)
	BaBORNet	100	4,651 (40)	4,597	1,091	1,155 (394)
	LaBORNet	200	4,457 (17)	**4,386**	1,897	4,450 (1,185)
	BaBORNet	200	4,620 (35)	4,597	1,508	2,290 (525)
	penalty EA	200	5,239 (210)	4,948	773	6,280 (1,248)
Turkey19	LaBORNet	100	2,348,899 (398,180)	1,886,350	6,300	8,250 (1,305)
	BaBORNet	100	1,670,062 (34,700)	1,620,210	23,210	8,315 (1,573)
	LaBORNet	200	2,160,075 (170,580)	1,802,870	25,200	15,460 (3,095)
	BaBORNet	200	1,650,683 (18,250)	**1,624,960**	50,090	13,980 (3,473)
	penalty EA	200	2,898,091 (263,880)	2,499,080	8,100	18,980 (1,603)
ger15	LaBORNet	100	49,284 (2,130)	46,326	3,390	5,480 (2,161)
	BaBORNet	100	45,832 (460)	45,004	23,408	7,245 (2,036)
	LaBORNet	200	47,419 (1,820)	44,648	10,800	11,130 (3,639)
	BaBORNet	200	45,101 (700)	**43,996**	45,400	12,510 (4,627)
	penalty EA	200	55,402 (1,900)	52,644	6,300	13,240 (2,045)
ger20	LaBORNet	100	58,732 (4,650)	51,064	10,950	8,985 (1,487)
	BaBORNet	100	52,553 (818)	51,064	23,580	7,945 (1,722)
	LaBORNet	200	57,980 (3,060)	53,998	26,100	13,370 (2,857)
	BaBORNet	200	51,277 (660)	**50,214**	79,620	17,110 (2,862)
	penalty EA	200	74,910 (10,000)	64,822	8,700	18,270 (2,747)
ger25	LaBORNet	100	91,064 (5,350)	84,620	14,520	10,000 (147)
	BaBORNet	100	59,077 (1,800)	56,046	24,780	8,580 (1,271)
	LaBORNet	200	76,750 (4,770)	70,396	45,370	19,530 (1,017)
	BaBORNet	200	56,718 (700)	**55,300**	90,345	17,480 (2,616)
	penalty EA	200	155,315 (50,700)	122,980	24,240	19,930 (395)
ger30	LaBORNet	100	171,649 (14,500)	158,664	21,750	9,030 (1,412)
	BaBORNet	100	91,177 (3,600)	84,548	24,960	9,585 (492)
	LaBORNet	200	147,674 (7,000)	138,646	44,040	18,830 (2,453)
	BaBORNet	200	83,968 (2,100)	**79,662**	102,300	19,540 (563)
	penalty EA	200	190,283 (25,153)	159756	84,600	33,329 (3,714)

topology and appropriate link options. When using BaBORNet, the EA only has to find a good topology and proper link options are determined by the local search strategy CURE.

Comparing the performance of LaBORNet and BaBORNet for different population sizes shows the expected behavior. For small problems, doubling the population size from *pop* = 100 to *pop* = 200 only slightly increases the solution quality as both population sizes allow to efficiently solve the problem. Contrastly, large problems like ger30 are more difficult and increasing the population size allows to increase the success probability of EAs (compare Harik et al.[14]) resulting in better results when using a higher population size.

Figure 5 compares the convergence behavior of the different EAs exemplary for the Deeter10 and ger30 problem. It shows the average best fitness over the number of generations for LaBORNet, BaBORNet, and the penalty EA. The plots are averaged over 10 runs and a population size *pop* = 200 is used. The figures reveal that due to the repair heuristic (local search) LaBORNet as well as BaBORNet find better solutions than the penalty EA in the initial population. Consequently, both repair



(a) Deeter10 (b) ger30

Fig. 5. $C(G_{best})$ over the number of generations for Deeter10 and ger30. We compare LaBORNet, BaBORNet, and a penalty EA ($pop = 200$).

approaches outperform the penalty approach. Furthermore, as already discussed, for small problems like Deeter10, LaBORNet outperforms BaBORNet, whereas for the larger ger30 problem BaBORNet finds better solutions than LaBORNet.

The remaining aspects are the running time t_{conv} and the number of fitness evaluations. The most time-consuming elements of the EA runs are the calculations of R_{All}. The number of calculations of R_{All} depends on the repair strategy CURE and on the solution that has to be repaired. The running time per fitness evaluation is expected to be higher for BaBORNet in comparison to LaBORNet as BaBORNet stores no information about link options. When using BaBORNet the repair heuristic has to determine all link types by iteratively increasing the reliability of links and it is necessary to calculate R_{All} in each iteration. When using LaBORNet, the link types are encoded in an individual and there are less repair steps (and less calculations of R_{All}) necessary to get a feasible solution.

Comparing the number of evaluations in table 1 shows about similar values for all three EA approaches. With increasing problem size n the number of evaluations increases until about 10,000 for $pop = 100$ and 20,000 for $pop = 200$ (the runs are stopped after 200 generations). Comparing t_{conv} to the number of evaluations reveals that the penalty EA is fastest whereas the repair approaches need much longer. The reason is that R_{All} is calculated only once for each solution if the penalty EA is used. When using repair approaches, an infeasible solutions must be repaired what makes it necessary to calculate R_{All} several times to repair one infeasible solution. The highest number of calculations of R_{All} are necessary for BaBORNet (resulting in a high running time t_{conv}) as the repair process of a solution always starts with the most unreliable link options and needs a large number of steps.

Although the computational effort of the repair approaches is higher when using the same population size, they outperform the penalty approach when using the same computational time. The numbers show that a penalty EA with $pop = 200$ needs about the same t_{conv} as repair approaches with $pop = 100$. Comparing the quality of the solutions that can be found using about the same running time t_{conv} reveals that the repair approaches, LaBORNet and BaBORNet, find much better solutions in comparison to the penalty EA when using about the same t_{conv}.

5. Summary and Conclusions

This paper presented two new approaches for the reliable communication network design (RCND) problem. The RCND problem assumes that links are unreliable and that for each link several options are available with different reliabilities and costs. The goal is to find a cost-minimal network design that satisfies a given overall reliability constraint. The paper presented in section 2 existing measurements for the overall reliability of a network and reviewed existing approaches for the RCND problem. In section 3, two new EA approaches, LaBORNet and BaBORNet, are presented. LaBORNet uses an encoding that encodes the network topology as well as the used link options and repairs infeasible solutions using the repair heuristic CURE. BaBORNet encodes only the network topology and determines the link options by using the repair heuristic CURE as a local search method. An investigation into the performance of the two repair approaches together with a comparison to a penalty EA approach from the literature is presented in section 4.

This presented results show that RCND problems can be solved more efficiently using the proposed EA approaches, LaBORNet and BaBORNet, than using penalty approaches. By using the new EA approaches, significantly better solutions for existing test problems from the literature can be found. The two new EA approaches using repair mechanisms outperform existing penalty approaches reliably and generate solutions of higher quality for all considered test problems. A direct comparison between the two new approaches reveals that BaBORNet, that means combining an EA who determines the network topology with a local search method that determines the link options, results in better solutions but needs more computational effort than LaBORNet, where the EA has to determine both the topology and the link options.

This work focus on reliability and cost aspects for network design problems. Real world design problems are often highly constraint problems. In a more complex planing scenario the capacity of nodes and links is limited by the technical properties and a set of commodities for the network is given. Future work might incorporate capacity issues in the design process. By adding capacity constraints for network nodes and links in the model the problem is extended to a other interesting research field considering routing and re-routing traffic in the network topology. Furthermore, we plan to examine the problem by a multiobjective approach that considers costs and reliability simultaneously.

References

1. J.M. Baldwin. A new factor in evolution. *American Naturalist*, 30:441–451, 1896.

2. B. Baran, S. Duarte, and D. Benítez. Telecommunication network design with parallel multi-objective evolutionary algorithms. In *IFIP/ACM Latin America Networking Conference*, La Paz, Bolivia, 2003.
3. B. Baran and F. Laufer. Topological optimization of reliable networks using a-teams. In *Proceedings of World Multiconference on Systemics, Cybernetics and Informatics - SCI '99 and ISAS '99*, volume 5, 1999.
4. L.T.M. Berry, B.A. Murtagh, G. McMahon, S. Sudgen, and L. Welling. An integrated GA-LP approach to communication network design. *Telecommunication Systems*, 12:265–280, 1999.
5. Y. Chen, J. Li, and J. Chen. A new algorithm for network probabilistic connectivity. In *Military Communications Conference Proceedings*, volume 2, pages 920–923, 1999.
6. Charles J. Colbourn. *The Combinatorics of Network Reliability*. Oxford University Press, 1987.
7. Darren L. Deeter and Alice E. Smith. Economic design of reliable networks. *IIE Transactions, Special Issue on Economics of Reliable Engineering*, 30:1161–1174, 1998.
8. B. Dengiz and C. Alabas. A simulated annealing algorithm for design of computer communication networks. In *Proceedings of World Multiconference on Systemics, Cybernetics and Informatics, SCI 2001*, volume 5, 2001.
9. B. Dengiz, F. Altiparmak, and A. E. Smith. Local search genetic algorithm for optimal design of reliable networks. *IEEE Transactions on Evolutionary Computing*, 1(3):179–188, 1997.
10. S. Duarte and B. Barán. Multiobjective network design optimisation using parallel evolutionary algorithms. In *Proccedings of XXVII Conferencia Latinoamericana de Informática CLEI'2001*, Merida, Venezuela, 2001.
11. G.S. Fishman. *Monte Carlo Simulation*. Springer-Verlag, 1995.
12. M. R. Garey and D. S. Johnson. *Computers and Intractibility: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, San Fransisco, 1979.
13. J. Gottlieb. *Evolutionary Algorithms for Constrained Optimization Problems*. unpublished PhD thesis, Technische Universität Clausthal, Institut für Informatik, Clausthal, Germany, 1999.
14. G. Harik, E. Cantú-Paz, D. E. Goldberg, and B. L. Miller. The gambler's ruin problem, genetic algorithms, and the sizing of populations. *Evolutionary Computation*, 7(3):231–253, 1999.
15. A. Konak and A. Smith. An improved general upperbound for all-terminal network reliability. Technical report, University of Pittsburg, 1998.
16. B. Liu and K. Iwamura. Topological optimization model for communication network with multiple reliability goals. *Computer and Mathematics with Applications*, 39:59–69, 2000.
17. E. Manzi, M. Labbe, G. Latouche, and F. Maffioli. Fishman's sampling plan for computing network reliability. *IEEE Transactions on Reliability*, 50(1):41–46, 2001.
18. Z. Michalewicz and D. B. Fogel. *How to solve it: modern heuristics*. Springer-Verlag, 2 edition, 2004.
19. L.C. Monticone. An implementation of the buzacott algorithm for network global-reliability. *IEEE Transactions on Reliability*, 42(1):46–49, 1993.
20. D. Orvosh and L. Davis. Using a genetic algorithm to optimize problems with feasibility constraints. *Proceedings of the First IEEE Conference on Evolutionary Computation*, 2:548–553, 1994.
21. D. Reichelt, F. Rothlauf, and P. Gmilkowsky. Designing reliable communication networks with a genetic algorithm using a repair heuristic. In *Proceedings 4th European Conference, EvoCOP 2004*, volume 3004 of *LNCS*, pages 177–187. Springer, 2004.

22. M. Stoer and F. Wagner. A simple min cut algorithm. In *Algorithms - ESA '94 Second Annual European Symposium*, pages 141–147, Berlin at al., 1994. Springer-Verlag.
23. D.L. Whitley, S.V. Gordon, and K. E. Mathias. Lamarckian Evolution, The Baldwin Effect and Function Optimization. In *Proceedings of Parallel Problem Solving from Nature - PPSN III*, volume 866 of *LNCS*, pages 6–15. Springer, 1994.
24. L. Ying. Analysis method of survivability of probabilistic networks. *Military Communication Technology Magazine*, 48, 1993.