Active Redundancy Allocation in Complex Systems by Using Different Optimization Methods

Petru Caşcaval and Florin Leon(✉)

Department of Computer Science and Engineering,
"Gheorghe Asachi" Technical University of Iaşi, Iaşi, Romania
{cascaval,florin.leon}@tuiasi.ro

Abstract. This paper addresses the issue of optimal allocation of spare modules in complex series-parallel redundant systems in order to obtain a required reliability under cost constraints. To solve this optimization problem, an analytical method based on the Lagrange multipliers technique is first applied. Then the results are improved by using other optimization methods such as an evolutionary algorithm and an original fine tuning algorithm based on the idea of hill climbing. The numerical results highlight the advantage of combining analytical approaches with fine tuning algorithms in case of very large systems. By using such a combined technique, better solutions are obtained than those given by classic heuristic search algorithms.

Keywords: Reliability allocation · Series-parallel reliability models · Lagrange multipliers · Evolutionary algorithms · Pairwise Hill Climbing

1 Introduction

The reliability design of a complex system is one of the most studied topics in the literature. The problems mainly refer to the kind of solution (reliability allocation and/or redundancy allocation), the kind of redundancy (active, standby, etc.), the type of the system (binary or multi-state), the levels of the redundancy (multi-level system or multiple component choice) etc. All these issues have practical applications and provide a good sphere for further research. An excellent overview of all these problems can be found in [1–4].

According to the decision variables [3, 4], a reliability optimization problem may belong to the following types: (a) reliability allocation, when the decision variables are component reliabilities, (b) redundancy allocation, when the variable is the number of component units, and (c) reliability-redundancy allocation, when the decision variables include both the component reliabilities and the redundancies.

In this paper we address the problem of redundancy allocation in which the number of redundant units in a series-parallel reliability model is the only decision variable. Unfortunately, as presented in [5], this is a problem that falls into the NP-hard category.

© Springer Nature Switzerland AG 2019
N. T. Nguyen et al. (Eds.): ICCCI 2019, LNAI 11683, pp. 625–637, 2019.
https://doi.org/10.1007/978-3-030-28377-3_52

We have limited ourselves to the binary systems in which each component is either operational or failed and, regarding the kind of redundancy, we focus only on the active spares.

To solve this optimization problem of redundancy allocation, more methods or techniques can be applied, such as intuitive engineering methods [6], heuristic search algorithms [7–10], analytical methods based on Lagrange multipliers [11–13], or dynamic programming [14]. Other metaheuristic methods based on genetic algorithms are also appropriate [15–18].

2 Notations and Preliminary Considerations

Reliability is the probability that a component or a system works successfully within a given period of time. A s*eries-parallel model* is a reliability model corresponding to a redundant system consisting of basic components and other active spare components.

The following notations are used throughout the paper: n is number of components in the non-redundant system or number of subsystems in the redundant system, as the case; r_i is the reliability of a component of type i, $i \in \{1, 2, \ldots, n\}$, for a given period of time; c_i is the cost of a component of type i; R_{ns} is the reliability of the non-redundant system (system with series reliability model); k_i is the number of components that compose the redundant subsystem i, $i \in \{1, 2, \ldots, n\}$; R_i is the reliability of subsystem i (subsystem with parallel reliability model); C_i is the cost of subsystem i; R_{rs} is the reliability of the redundant system (system with series-parallel reliability model); C_{rs} is the cost of the redundant system; R^* is the required reliability level for the system; C^* is the maximum accepted cost for the system.

It is assumed that: each component in the system is either operational or failed, i.e. a binary system, a spare is identical with the basic component, and the events of failure that affect the components of the system are stochastically independent.

3 Problem Description

Let us consider a non-redundant system composed of n basic components for which the reliability model is a series one as presented in Fig. 1. For complex systems with a large number of components, the reliability of the system without redundancy is often quite low. To reach a required reliability, spare components are added so that the reliability model for this redundant system is a series-parallel one as presented in Fig. 2.

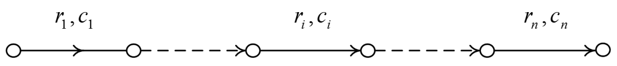

Fig. 1. Series reliability model for a non-redundant system

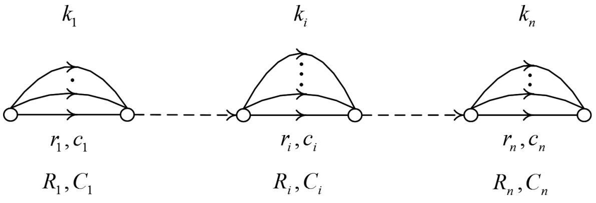

Fig. 2. Series-parallel reliability model for a redundant system with spare components

Typically, in this allocation process the criterion is reliability, cost, weight or volume. One or more criteria can be considered in an objective function, while the others may be considered as constraints [3, 4, 19]. From a mathematical point of view, one must solved an optimization problem of an objective function with constraints.

In this work, we address the issue of maximizing system reliability within the cost constraint. Thus, one must maximize the reliability function $R_{rs} = \prod_{i=1}^{n} R_i$, with the restriction of cost $\sum_{i=1}^{n} C_i \leq C^*$.

In case of large systems, to master the complexity of the problem, we choose to first apply an analytical method based on Lagrange multipliers in order to quickly obtain an approximate solution, and then, this approximate solution is improved by using other optimization methods such as an evolutionary algorithm and an original fine tuning algorithm. To check the results, we have also implemented some heuristic algorithms, as presented in the following section.

4 Heuristic Search Algorithms

For the reliability model presented in Fig. 2, the following equation is valid: $R_{rs}(t) = \prod_{i=1}^{n} R_i(t) \leq \min_i\{R_i(t)\}$. Starting from this observation, the following two heuristic methods are applied to solve this optimal allocation problem.

Algorithm 1: This is a greedy algorithm given by Misra [7] that tries to make an optimal choice at each step. Thus, starting with the minimum system design presented in Fig. 1 ($k_i, i \in \{1, 2, \ldots, n\}$), the system reliability is increased by adding one component to the subsystem with the lowest reliability. This process is repeated as long as the cost constraint is met.

Algorithm 2: This algorithm given by Rajendra Prasad, Nair, and Aneja [9] ensures an acceleration of the allocation process. The basic idea is that the subsystem with the highest reliability will have the smallest number of components, and the least reliable subsystem, the greatest number of components. Thus, starting with the system in a non-redundant form, the reliability is increased by adding one component to each subsystem (a row allocation) as long as the cost constraint is met. For the most reliable subsystem,

this is the final allocation. The process of allocation continues in the same manner with the other subsystems, until no allocation is possible any longer.

As presented in Sect. 8, these heuristic algorithms are useful in many cases, but sometimes they do not give good solutions. Therefore, we choose to apply first an analytical method based on Lagrange multipliers as presented in the following section.

5 Analytical Approach

As the spares are identical with the basic component, for the series-parallel reliability model presented in Fig. 2, the reliability function can be expressed by the equation:

$$R_{rs} = \prod_{i=1}^{n} \left(1 - (1 - r_i)^{k_i}\right). \tag{1}$$

We have to determine the values k_i that maximize the function R_{rs} with the cost constraint:

$$\sum_{i=1}^{n} c_i k_i \le C^*. \tag{2}$$

Based on (1) and (2), the following Lagrangian function results:

$$L(k_1, k_2, \ldots, k_n, \lambda) = \prod_{i=1}^{n} \left(1 - (1 - r_i)^{k_i}\right) + \lambda \left(\sum_{i=1}^{n} c_i k_i - C^*\right), \tag{3}$$

where λ is the Lagrange multiplier.

Thus, instead of maximizing the function R_{rs} given by (1) within the cost restriction (2), we have to maximize the Lagrange function given by (3) without constraints. For this propose, a system with partial derivatives must be solved, where $\partial L / \partial k_i = 0,\ i \in \{1, 2, \ldots, n\}$, and $\partial L / \partial \lambda = 0$. But the resulting system of algebraic equations is very difficult to solve because of the products that appear. For this reason, we use another way of expressing the reliability of the system, as presented as follows.

The reliability of a system can be expressed by the reliability function (R) or the non-reliability function ($F = 1 - R$), that means, by a point within the unit segment [0, 1]. As presented in Fig. 3, this value can be uniquely identified by a point on the diagonal of a square with the side length equal to the unit (denoted by P).

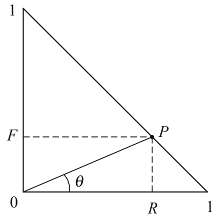

Fig. 3. Another way of expressing the reliability

The point P in Fig. 3 is uniquely identified by the angle θ, and thus by $x = \tan(\theta)$. Related to the two reliability indicators, R and x, the following equations are valid: $x = (1 - R)/R = 1/R - 1$, and $R = 1/(x + 1)$. Notice that, if $R \to 1$ then $x \to 0$, and if $R \to 0$ then $x \to \infty$.

Let us now consider a system with a series reliability model, composed of n modules of reliability given by x_i. The system reliability, denoted by $x_{S(n)}$, can be expressed by the following equation:

$$x_{\mathrm{S}(n)} = \frac{1}{R_{S(n)}} - 1 = \frac{1}{\prod_{i=1}^{n} \frac{1}{x_i + 1}} - 1 = \prod_{i=1}^{n} (x_i + 1) - 1. \tag{4}$$

This relationship is rather complicated, but for the values $x_i \ll 1$, $i \in \{1, 2, \ldots, n\}$, the products that are formed can be neglected and the reliability of the system can be approximated with good accuracy by the equation $x_{S(n)} \approx \sum_{i=1}^{n} x_i$. In the same manner, one show that for a parallel reliability model with n components, when $x_i \ll 1$, $i \in \{1, 2, \ldots, n\}$ the reliability of the redundant system denoted by $x_{P(n)}$ can be approximated with good accuracy by $x_{P(n)} \approx \prod_{i=1}^{n} x_i$.

In this form of representation for the system reliability, the optimization problem assumes the maximization of the reliability function $x_{rs} = \sum_{i=1}^{n} x_i^{k_i}$ with the cost constraint (2). In this case, the Lagrangian function is:

$$L(k_1, k_2, \ldots k_n, \lambda) = \sum_{i=1}^{n} x_i^{k_i} + \lambda \left(\sum_{i=1}^{n} c_i k_i - C^* \right). \tag{5}$$

By applying the partial derivatives, and using the notations $\alpha_i = 1/\ln x_i$ and $\beta_i = \alpha_i (\ln c_i + \ln(-\alpha_i))$, $i \in \{1, 2, \ldots, n\}$, the system solution of algebraic equations is:

$$\lambda = e^{\gamma}, \text{ where } \gamma = \left(C^* - \sum_{i=1}^{n} c_i \beta_i \right) \Big/ \sum_{i=1}^{n} c_i \alpha_i, \text{ and } k_i = \alpha_i \ln \lambda + \beta_i, \\ i = 1, 2, \ldots, n \tag{6}$$

Note that, this analytical result is valid only if $\alpha_i < 0$ (i.e. $r_i > 0.5$).

The only impediment for solving the problem remains that the values k_i obtained by applying (6) are real values, and the allocation is, by its nature, a discrete problem. Therefore, it is necessary to determine the optimal allocation in integer numbers starting from the actual solution.

An approximate solution results by adopting the integers closest to the real values. But, in this case, it is possible that the new solutions for k_i, $i \in \{1, 2, \ldots, n\}$, not longer satisfy the cost constraint. Starting from the value of the Lagrange multiplier λ

given by (6), a searching process is carried out in a neighborhood of this value in order to obtain a better approximate solution for k_i, $i \in \{1, 2, \ldots, n\}$, while satisfying the cost constraint. Unfortunately, this approximate solution is not accurate enough. Consequently, this approximate solution is further improved by applying other methods of refining the search process, as shown in the following sections.

As mentioned in Sect. 1, regarding the kind of redundancy, in this work we have limited ourselves to the active spares, because for the case of passive spares the Lagrangian function is more complicated and the system of partial derivatives is very difficult to solve. But the following optimization methods may be tailored to cover this case as well, or better, to cover the most general case in which some subsystems may have active spares and the others, passive spares.

6 Evolutionary Algorithm

Evolutionary algorithms are optimization methods inspired by natural selection. The search is made in parallel, with a population of potential solutions, also known as individuals or chromosomes, which are randomly initialized. Each chromosome is associated with a value determined by the fitness function that describes the problem to be solved. The evolution of the population is based on the successive application of genetic operators such as selection, which gives the better individuals more changes to reproduce, crossover, which combines the genetic information of two parents and creates offspring, and mutation, which may slightly change a newly generated child in order to introduce new genetic information into the population. The search progresses until a stopping criterion is met.

In our case, the encoding of the problem is real-valued, and each chromosome has n real genes, corresponding to k_i. The domain of the genes $[1, k_{max}]$ depends on the problem. k_{max} is the maximum estimated value for any k_i.

The fitness function is the expression in (1). Since the genes are real-values and k_i are integer numbers, the gene values are rounded in order to convert them into integers.

Since we have a constraint, one method of handling it is to add a penalty to the fitness function of the chromosomes that do not satisfy the constraint. However, this can have a very negative impact on the quality of the solution, because, especially in the beginning, very few or no individuals may satisfy the cost constraint. This can likely lead to very poor local optima, or may cause the algorithm to fail altogether in finding any solution.

The method adopted in the present work was to repair the chromosomes that violate the cost constraint, i.e. randomly removing one redundant component at a time, until the remaining components satisfy the constraint. Each subsystem must have at least one component, so only redundant components may be removed.

The selection method is tournament selection with two individuals. We use arithmetic crossover and mutation by gene resetting. As a stopping criterion, both fitness convergence and a fixed number of generation were tried. It was found that after a small number of generation, the values of all the fitness functions become very close, but these small differences actually make a big difference for the quality of the final result. Thus, a fixed number of generations was chosen as a stopping criterion.

Several variants of genetic evolution were evaluated:

- the typical method that generates offspring from parents, creates a complete new generation and then the new generation replaces the old one;
- elitism, where the best individual of a generation is directly copied into the next one, and that guarantees that the best solution at one time is never lost;
- steady state evolution, where from the current population of size s a new population of size s is generated by regular selection, crossover and mutation, but the new population does not directly replace the old one; instead, the two populations are merged, and from the resulting population of size $2s$, the best s individuals are selected to form the next generation;
- optimized offspring generation, where after a child has been created by regular selection, crossover and mutation, it is not directly inserted into the next generation; instead the individual with the best fitness out of the mother, the father and the child is inserted into the next generation.

From the experiments conducted for large problem instances, it was found that the optimized offspring generation combined with elitism provided the best results.

7 Pairwise Hill Climbing

An evolutionary algorithm has a good chance of find the global optimum, but it often finds a solution within the neighborhood of the optimum. On the other hand, the analytical method presented in Sect. 5 provides an approximate solution because of the conversion of the real solution into an integer one. In both situations, it is important to fine tune the initial approximation, it order to find a better solution, closer or equal to the actual optimum.

The typical hill climbing procedure starts from an initial solution and generates its neighbors in the problem space. The best neighbor is identified and if the value of its objective function is better than the one of the initial solution, the neighbor becomes the current solution and the same process is repeated until no neighbor is better that the current (and final) solution.

However, for the problems we study, the above algorithm cannot be applied because of the existence of the constraint. The approximate solvers usually find good, but sub-optimal solutions, which all have a cost very close or equal to C^*. Therefore, the classic hill climbing procedure cannot increase reliability further by adding, for example, an additional component, because the cost constraint would be violated.

In order to increase reliability at fairly the same total cost, a few, e.g. usually 1–2, components need to be removed from one or a few redundant subsystems, while simultaneously a few components need to be added to other few redundant subsystems.

Therefore, beside the addition of a component, a swap operation must also be considered when generating the neighbors of an allocation.

In some situations, the initial solution may not be in a direct neighborhood of the optimum, and thus solely relying on the addition of new components, which is the only way to increase reliability, in not sufficient.

Consequently, we propose an original fine tuning algorithm, Pairwise Hill Climbing, presented in Pseudocode 1, based on the main idea of hill climbing, but with several heuristics that make it more appropriate for our class of problems.

```
Pairwise Hill Climbing(initial)
  solutionList.Add(initial)

  for i in [1..n] do
    initial2 ← initial
    initial2.k[i] ← max(1, initial2.k[i] - 1)
    solutionList.Add(initial2)

  initial3 ← initial
  for i in [1..n] do
    initial3.k[i] ← max(1, initial3.k[i] - 1)
  solutionList.Add(initial3)

best ← initial
while solutionList is not empty do
  bestSoFar ← solutionList[bi] where
    solutionList[bi].reliability is the largest
  solutionList.Remove(bestSoFar)
  Refine(bestSoFar)

return best
```

```
Refine(current)
  for each pair (i, j) in ([1.. n], [1.. n]) and i ≠ j do
    next ← current with (next.k[i], next.k[j]) += (1, 0)
      and then += (1, -1)
    next.reliability ← ComputeReliability ()

    if IsFeasible(next) and next.reliability >
      current.reliability then
      solutionList.Add(next)

  if current.reliability > best.reliability then
    best ← current
```

Pseudocode 1. Pairwise Hill Climbing

As the pseudocode shows, the successive allocations, starting from $n + 2$ initial points, are added to a priority queue, which helps to first expand the most promising solution. However, if the corresponding subtree yields a suboptimal solution, other parts of the tree can be used to find a better one. From the practical point of view, the search may be limited to a maximum number of levels in the tree l_{max} and to a maximum number of solutions taken from the queue s_{max}.

8 Experimental Results

In this section, we present the results of the previously described methods for three problems, designed in such a way as to make the optimization more difficult. Thus, for some components, we have chosen values so that, for a very similar reliability, the cost is very different. For example, in Problem 3, as highlighted by bold face in Table 1, the components with order number 4 and 43 have quite similar reliabilities (0.74 and 0.75, respectively), but very different cost values (49 and 4, respectively). The same remark holds for components with order number 39 and 44. The parameters of the components are presented in Table 1. For all three problems, $n = 50$.

The obtained results are presented in Table 2.

Methods 1 and 2 are the heuristic methods presented in Sect. 4.

Method 3 is the analytical method from Sect. 5 where the real-valued coefficients are converted into integers by rounding them down, i.e. $k_i = \lfloor k_i^r \rfloor$. This method obviously provides suboptimal results, but it can be a good starting point for solution improvement, because its solution has a good chance of being in the neighborhood of the optimal one.

Table 1. The optimization problems considered in the experimental study

	Component reliability $(r_1, r_2, \ldots, r_n)$	Component cost $(c_1, c_2, \ldots, c_n)$	C^*
Problem 1	0.59, 0.70, 0.84, 0.99, 0.81, 0.69, 0.88, 0.68, 0.65, 0.92, 0.67, 0.73, 0.68, 0.87, 0.71, 0.51, 0.66, 0.98, 0.73, 0.88, 0.87, 0.55, 0.73, 0.55, 0.59, 0.83, 0.76, 0.58, 0.77, 0.52, 0.87, 0.76, 0.77, 0.71, 0.63, 0.96, 0.97, 0.96, 0.79, 0.63, 0.61, 0.88, 0.86, 0.71, 0.91, 0.91, 0.93, 0.82, 0.72, 0.93	14, 5, 21, 21, 9, 9, 44, 35, 27, 30, 16, 22, 28, 22, 7, 15, 33, 47, 13, 38, 37, 34, 11, 41, 9, 45, 35, 48, 34, 41, 7, 17, 21, 10, 2, 33, 9, 40, 23, 38, 4, 34, 32, 20, 17, 40, 26, 48, 4, 32	6240
Problem 2	0.83, 0.91, 0.88, 0.99, 0.93, 0.73, 0.89, 0.96, 0.80, 0.95, 0.60, 0.88, 0.64, 0.83, 0.70, 0.86, 0.95, 0.70, 0.85, 0.80, 0.66, 0.86, 0.86, 0.84, 0.72, 0.91, 0.62, 0.69, 0.78, 0.70, 0.83, 0.86, 0.92, 0.53, 0.58, 0.65, 0.77, 0.57, 0.93, 0.79, 0.78, 0.67, 0.75, 0.94, 0.72, 0.70, 0.91, 0.97, 0.63, 0.80	47, 25, 21, 49, 20, 13, 44, 28, 8, 23, 45, 44, 34, 7, 14, 15, 42, 25, 42, 29, 23, 44, 2, 22, 7, 17, 18, 28, 20, 26, 48, 21, 8, 5, 17, 2, 6, 32, 49, 50, 26, 22, 4, 4, 42, 31, 44, 10, 45, 26	6370
Problem 3	0.83, 0.91, 0.88, **0.74**, 0.93, 0.73, 0.89, 0.96, 0.80, 0.95, 0.60, 0.88, 0.64, 0.83, 0.70, 0.86, 0.95, 0.70, 0.85, 0.80, 0.66, 0.86, 0.61, 0.84, 0.72, 0.91, 0.62, 0.69, 0.78, 0.70, 0.83, 0.86, 0.92, 0.53, 0.58, 0.56, 0.77, 0.57, **0.89**, 0.79, 0.78, 0.67, **0.75**, **0.90**, 0.72, 0.70, 0.91, 0.97, 0.55, 0.80	47, 25, 21, **49**, 20, 13, 44, 28, 8, 23, 45, 44, 34, 7, 14, 15, 42, 25, 42, 29, 23, 44, 2, 22, 7, 17, 18, 28, 20, 26, 48, 21, 8, 5, 17, 2, 6, 32, **49**, 50, 26, 22, **4**, **4**, 42, 31, 44, 10, 45, 26	6370

Method 4 involves an additional heuristic search starting from the initial estimates of the analytical method.

Method 5 uses the same estimates given by the analytical method, but employs the Pairwise Hill Climbing (PHC) search presented in Sect. 7. For PHC, the parameters used are l_{max} = 20 and s_{max} = 1000.

Method 6 only uses the Evolutionary Algorithm (EA) described in Sect. 6 with a large number of generations, i.e. 10000. In all cases, the other parameters of the EA are the following: the population size is 100, the crossover rate is 0.9, the mutation rate is 0.1. The search range for this method is 1–15 for each gene, i.e. k_i. Since the EA is nondeterministic, 10 runs for each configuration were performed and the best result was reported.

Method 7 uses the EA with a smaller number of generations, i.e. 1000, and its solution is used as the initial point for the PHC search.

Finally, method 8 uses the solution of the analytical method to define the search range for the EA genes. The range of a gene is the suboptimal $k_i \pm 2$, but ensuring that the lower limit remains at least 1.

Beside the actual reliability value of the redundant system R_{rs}, we included an additional, more intuitive measure of the results, namely the redundancy efficiency defined as: $Ef = (1 - R_{ns})/(1 - R_{rs})$. The efficiency shows how many times the risk of a failure decreases for the redundant system, compared to the baseline, non-redundant one. For example, at the first glance, it may not seem to be a big difference between two reliability values of 0.88 and 0.98, e.g. for Problem 2. However, the efficiency shows us that the risk of a failure decreases almost 6 times.

Table 2. The results obtained (R_{rs}, Ef, C_{rs}) with different optimization methods

Method	Problem 1	Problem 2	Problem 3
1. Heuristic 1	0.8242, 5.69, 6239	0.9803, 50.76, 6369	0.9701, 33.44, 6370
2. Heuristic 2	0.9628, 26.88, 6240	0.8882, 8.94, 6370	0.8591, 7.10, 6370
3. Analytical method with k_i rounded down	0.9678, 31.06, 6204	0.9830, 58.82, 6344	0.9755, 40.82, 6351
4. Analytical method followed by heuristics	0.9685, 31.75, 6239	0.9831, 59.17, 6369	0.9757, 41.15, 6369
5. Analytical method followed by pairwise hill climbing	$\underline{0.9690}$, $\underline{32.26}$, 6239	$\underline{0.9835}$, $\underline{60.77}$, 6370	$\underline{0.9759}$, $\underline{41.50}$, 6370
6. Evolutionary algorithm with long search	0.9687, 31.99, 6239	0.9819, 55.42, 6370	0.9742, 38.82, 6370
7. Evolutionary algorithm followed by pairwise hill climbing	0.9668, 30.19, 6240	0.9825, 57.41, 6370	0.9751, 40.27, 6369
8. Analytical method followed by evolutionary algorithm with reduced search range	0.9686, 31.87, 6239	0.9834, 60.28, 6370	0.9755, 40.89, 6370

As one can notice, the first two heuristic methods give very different results for all the three problems. This highlights the difficulty of the optimization problems we considered.

The first observation is that in all three cases, the best results are obtained by the hybrid Method 5. The results of the analytical method are a very good starting point for the PHC, because the component allocation is likely less than the optimal one and a procedure based on the hill climbing idea can easily add new components and thus improve the objective function.

Conversely, it is more difficult for the PHC to improve more extensively the EA solution, because that is already near the maximum cost, and therefore some components need to be removed in order to add others, while keeping the solution in the feasibility region, i.e. satisfying the cost constraint. As it was seen, such refining does not only address simple component swaps, but sometime, more components should be removed from or added to a subsystem, and the changes to be made involve not only two subsystems, but often more.

It is somewhat surprising that the EA does not seem capable of finding the optimum, even with a very large number of generations and even starting from an estimated neighborhood of the optimal solution. This is because the problem has many local optima with the maximum cost, and the genetic search procedure is also affected by the fact that infeasible solutions are repaired, i.e. modified outside the influence of the genetic operators, by randomly removing components until the cost constraint is met. This repairing procedure is however critical to the convergence of the EA; if one uses penalties for constraint violation, usually the EA does not find a solution at all, or finds a very poor one. Thus, a different repairing procedure may be needed, where components are not removed at random, but some problem specific information is used in order to find the most appropriate change that can help the optimization process.

It is also important to discuss the execution times of these methods. The first four techniques are very fast, because they are based on the application of formulas or a linear search in a very limited range, comparable with n. The average execution times for ten runs for each problem using a computer with a 4-core 2 GHz Intel processor and 8 GB of RAM are given as follows: Method 5 – 1061.98 ms, Method 6 – 45,603.83 ms, Method 7 – 6,877.77 ms, Method 8 – 4252.38 ms.

Therefore, Method 5, which gives the best results, is also the fastest.

In Table 3, the best solutions are presented for each of the three problems under study.

Table 3. The best solutions found, i.e. the k_i values and total cost for the best R_{rs}

	Optimal allocation: $k_1, k_2, \ldots, k_n$	C_{rs}
Problem 1	8, 7, 4, 2, 5, 7, 3, 6, 7, 3, 7, 6, 6, 4, 7, 10, 6, 2, 6, 4, 4, 8, 6, 8, 9, 4, 5, 7, 5, 8, 5, 5, 5, 6, 9, 3, 3, 3, 5, 7, 9, 4, 4, 6, 4, 3, 3, 4, 7, 3	6239
Problem 2	4, 4, 4, 2, 3, 7, 4, 3, 6, 3, 8, 4, 7, 5, 7, 5, 3, 6, 4, 5, 7, 4, 6, 5, 7, 4, 8, 7, 6, 6, 4, 4, 4, 12, 9, 10, 7, 8, 3, 5, 5, 7, 7, 4, 6, 6, 3, 3, 7, 5	6370
Problem 3	4, 4, 4, 5, 3, 6, 3, 3, 5, 3, 7, 4, 7, 5, 7, 4, 3, 6, 4, 5, 7, 4, 11, 4, 7, 4, 8, 6, 5, 6, 4, 4, 4, 11, 9, 12, 6, 8, 3, 5, 5, 7, 7, 4, 6, 6, 3, 3, 8, 5	6370

9 Conclusions

In this paper several optimization methods where presented for the problem of maximizing the reliability of a redundant system, with a series-parallel reliability model, in the presence of cost-related constraints. Beside domain-specific heuristics and an analytical solution based on Lagrange multipliers, other general techniques were applied: an evolutionary algorithm with chromosome repairs meant to satisfy the constraint and an optimized offspring generation, and also a original algorithm based on the hill-climbing idea, but using swaps and a priority queue in addition to the incremental greedy improvements, in order to fine tune the approximate solutions found by other methods.

Our study shows that the hybrid methods provided the best results, especially the combination of the analytical method with the original Pairwise Hill Climbing algorithm, which is also the fastest.

As future directions of investigation, other optimization methods will be tried, and other variants of the problem will be addressed, for example, the issue of passive redundancy allocation and the converse problem of minimizing the cost while having a minimum imposed reliability, as a means to further verify the optimality of the solutions.

References

1. Tillman, F.A., Hwang, C.L., Kuo, W.: Optimization techniques for system reliability with redundancy. A review. IEEE Trans. Reliab. **26**, 148–155 (1977)
2. Tillman, F.A., Hwang, C.L., Kuo, W.: Optimization of System Reliability. Marcel Dekker, New York (1980)
3. Shooman, M.: Reliability of Computer Systems and Networks, pp. 331–383. Wiley, New York (2002)
4. Misra, K.B. (ed.): Handbook of Performability Engineering, pp. 499–532. Springer, London (2008). https://doi.org/10.1007/978-1-84800-131-2
5. Chern, M.: On the computational complexity of reliability redundancy allocation in series system. Oper. Res. Lett. **11**, 309–315 (1992)
6. Albert, A.A.: A measure of the effort required to increase reliability. Technical report No. 43, Stanford University, Applied Mathematics and Statistics Laboratory (1958)
7. Misra, K.B.: A simple approach for constrained redundancy optimization problems. IEEE Trans. Reliab. **R-20**(3), 117–120 (1971)
8. Agarwal, K.K., Gupta, J.S.: On minimizing the cost of reliable systems. IEEE Trans. Reliab. **R-24**, 205–206 (1975)
9. Rajendra Prasad, V., Nair, K.P.K., Aneja, Y.P.: A heuristic approach to optimal assignment of components to parallel-series network. IEEE Trans. Reliab. **40**(5), 555–558 (1992)
10. El-Neweihi, E., Proschan, F., Sethuraman, J.: Optimal allocation of components in parallel-series and series-parallel systems. J. Appl. Probab. **23**(3), 770–777 (1986)
11. Everett, H.: Generalized Lagrange multiplier method of solving problems of optimal allocation of resources. Oper. Res. **11**, 399–417 (1963)

12. Misra, K.B.: Reliability optimization of a series-parallel system, part I: Lagrangian multipliers approach, part II: maximum principle approach. IEEE Trans. Reliab. **R-21**, 230–238 (1972)
13. Blischke, W.R., Prabhakar, D.N.: Reliability: Modelling, Prediction, and Optimization. Wiley, New York (2000)
14. Belmann, R., Dreyfus, S.: Dynamic programming and the reliability of multi-component devices. Oper. Res. **6**(2), 200–206 (1958)
15. Coit, D.W., Smith, A.E.: Reliability optimization of series-parallel systems using a genetic algorithm. IEEE Trans. Reliab. **45**, 254–260 (1996)
16. Marseguerra, M., Zio, E.: System design optimization by genetic algorithms. In: Proceedings of the Annual Reliability and Maintainability Symposium, Los Angeles, CA, pp. 222–227. IEEE (2000)
17. Agarwal, M., Gupta, R.: Genetic search for redundancy optimization in complex systems. J. Qual. Maintenance Eng. **12**(4), 338–353 (2006)
18. Romera, R., Valdes, J.E., Zequeira, R.I.: Active redundancy allocation in systems. IEEE Trans. Reliab. **53**(3), 313–318 (2004)
19. Shooman, M.L., Marshall, C.: A mathematical formulation of reliability optimized design. In: Henry, J., Yvon, J.P. (eds.) System Modelling and Optimization. LNCIS, vol. 197, pp. 941–950. Springer, Heidelberg (1994). https://doi.org/10.1007/BFb0035543