This item was submitted to Loughborough's Institutional Repository (https://dspace.lboro.ac.uk/) by the author and is made available under the following Creative Commons Licence conditions.

creative commons

COMMONS DEED

Attribution-NonCommercial-NoDerivs 2.5

You are free:

- to copy, distribute, display, and perform the work

Under the following conditions:

Attribution. You must attribute the work in the manner specified by the author or licensor.

Noncommercial. You may not use this work for commercial purposes.

No Derivative Works. You may not alter, transform, or build upon this work.

- For any reuse or distribution, you must make clear to others the license terms of this work.
- Any of these conditions can be waived if you get permission from the copyright holder.

Your fair use and other rights are in no way affected by the above.

This is a human-readable summary of the Legal Code (the full license).

Disclaimer

For the full text of this licence, please go to:
http://creativecommons.org/licenses/by-nc-nd/2.5/

Int. J. Reliability and Safety, Vol. X, No. Y, XXXX

System design optimisation involving phased missions

Dovile Astapenko and Lisa M. Bartlett*

Department of Aeronautical and Automotive Engineering,
Loughborough University,
Leicestershire, LE11 3TU, UK
Email: D.Astapenko@lboro.ac.uk
Email: L.M.Bartlett@lboro.ac.uk
*Corresponding author

Abstract: The performance of a phased mission is defined as a succession of non-overlapping phases that constitute towards a continuous mission. The focus of this paper is to develop a method to construct an optimal design structure for a phased mission system when available resources are restricted and to ensure a minimal system failure probability throughout the whole mission. The implemented optimisation method employs fault tree analysis to represent the causes of failure in the system for each phase. Binary decision diagrams are used to quantify the failure probability of each phase and the whole mission, and a single objective genetic algorithm is chosen to solve the optimisation problem. Analysis of the optimisation process of a military vessel design during a training mission is presented and the obtained results are discussed.

Keywords: phased missions; optimisation; fault tree analysis; binary decision diagrams; genetic algorithms.

Reference to this paper should be made as follows: Astapenko, D. and Bartlett, L.M. (XXXX) 'System design optimisation involving phased missions', *Int. J. Reliability and Safety*, Vol. X, No. Y, pp.xxx–xxx.

Biographical notes: Dovile Astapenko is a PhD student in the Department of Aeronautical and Automotive Engineering at Loughborough University. She received her BSc degree in Applied Mathematics at Kaunas University of Technology (Lithuania) in 2003. She was also awarded an MSc degree in Applied Mathematics with distinction at Kaunas University of Technology in 2005. Her current research is focused in automated systems and phased mission design optimisation problems.

Lisa M. Bartlett is a Lecturer in the Department of Aeronautical and Automotive Engineering at Loughborough University. She is one of five academic members of staff within the Reliability Research Group of the department. She gained her PhD degree in fault tree analysis methods in 2000 from Loughborough University. Her PhD research focused on the Binary Decision Diagram approach, an alternative analysis method for fault tree analysis. Her current research interests are in safety system design optimisation, binary decision diagrams and fault diagnostics.

Copyright © 200X Inderscience Enterprises Ltd.

1 Introduction

A phased mission system represents a system whose performance is divided into consecutive non-overlapping phases. As a classical example an aircraft mission can be chosen where the mission consists of three phases: take-off, cruise and landing. It is assumed that the aircraft completes the mission if the tasks of each phase have been completed successfully. Failure to complete any phase results in the mission failure.

Methods used for phased mission reliability problems are grouped in two major categories. Markov models are usually employed to solve repairable phased mission system problems. They are also used for the analysis of systems where dependencies between component failures of mission phases exist, as discussed by Alam and Al-Saggaf (1986) and Kim and Park (1994). Fault Tree Analysis (FTA) is usually employed to solve non-repairable phased mission system reliability problems. A pioneering analysis of phased missions when employing fault trees was developed by Esary and Ziehms (1975). La Band and Andrews (2004) introduced the approach to analyse fault trees for phases using a modularisation method. It increased the efficiency of the analysis. Ma and Trivedi (1999) implemented the Binary Decision Diagram (BDD) methodology for analysis of the fault tree, which also made the analysis of phased mission systems computationally more efficient.

Despite the number of approaches for the evaluation of phased mission system reliability that have been developed, their application to phased mission system reliability optimisation problems has not been widely considered. There is limited demonstrated evidence in the literature for research that focuses on these optimisation problems. Susova and Petrov (1997) proposed a model for the optimisation of an aircraft system. The model is based on a Markov homogeneous process and is employed to ensure aircraft safety and minimise operation costs.

A large number of systems which appear in industry can be analysed as phased mission systems. Failure of any system, including a phased mission system, can have critical and even life-threatening consequences and it always results in financial losses. On the other hand, the improvement of system safety can increase costs and/or necessitate the use of other resources. The relevance of minimising the phased mission failure probability and optimising the use of limited resources at the same time becomes evident.

A phased mission optimisation problem is considered in the paper. The objective of the optimisation process is to replace a number of system components with alternative components in order to minimise the system failure probability throughout the mission. The improvements of the system are considered to be limited due to predefined design constraints and resources.

A phased mission system design structure is analysed employing fault trees for each phase where basic events are associated with system components. It means that changes in system design result in the alterations of at least one mission fault tree. While FTA is useful to analyse the construction of a system design, BDD-based methods are considered to be mathematically more efficient for mission failure analysis. Therefore, an approach presented by Prescott et al. (2008) was employed to find phased mission failure probability values. In the actual optimisation process the failure probabilities were employed as objective function values. A Single Objective Genetic Algorithm (SOGA) was chosen as the optimisation technique. A penalty function was implemented in order to incorporate predefined constraints on the available system resources into the optimisation problem.

2 Phased mission design optimisation algorithm

2.1 Application of fault tree and BDD analysis

One of the principal methodologies used in the discussed algorithm is FTA. The preparation stage of the algorithm requires each phased mission phase to be presented with individual fault trees. It enables both the listing of all system components and the identification of system failure causes which is then used to evaluate the system failure probability during the mission.

Basic events of a fault tree are factors that cause system failure when they appear on their own or in different combinations with other failure factors. Internal factors that cause system failure are system component failure modes. It means that a fault tree can also be used to represent the system design structure. If any system component is substituted with more than one additional component, it results in a new system design structure. A system fault tree will also be altered in order to include failure modes of the new system components. If failure of a component does not cause system failure, it does not appear as a basic event in the system fault tree and, therefore, it will not be used when evaluating the system failure probability. It follows that such a system component can be eliminated from the optimisation analysis. Following this, it is assumed that a system design structure comprises only those components which appear as basic events in the system fault tree.

Usually, in trying to improve system performance and/or to minimise its failure probability, the system design is modified by replacing a certain number of components. Existing system components can be replaced with a different number of redundant components and/or components of different types, i.e. components that have different characteristics. Therefore, the notation 'design variables' is introduced. Values of design variables identify the numbers of redundant components, redundancy types and types of new components used in the analysis. Each replaceable component can be associated with more than one design variable and every design variable must have more then two different values.

House events are introduced in the fault trees of system phases in order to implement their alterations due to changes in system structure when certain system components are replaced. The methodology of introducing house events associated with design variables into fault trees was discussed by Pattison (1999). Using this methodology one fault tree for each phase can be constructed despite a number of introduced design variables and varying numbers of their possible values presented in the problem. Otherwise, every single component replacement would require a new fault tree for every phase where at least one basic event of the associated component appears. It means that the number of fault trees would increase dramatically even if just a few design variables with various possible values were introduced. Consequently, the evaluation of mission failure probability would become rather complicated.

When using house events the resulting fault tree combines all possible design variations of a system structure. A fault tree for each corresponding individual system design can be derived by setting certain house events to 1 and keeping values of the rest of the house events set to 0. A house event is set to 1 if a logic event, associated with this house event, occurs and the house event is assigned the value 0 if the logic event does not occur. The fault trees for phases are then simplified by eliminating branches where values of house events are 0.

There are a number of methods that can be employed to evaluate the mission failure probability were mission phases are represented using fault trees. As mentioned earlier, in the proposed optimisation algorithm, a methodology introduced by Prescott et al. (2008) was chosen. Their approach provides failure probabilities for each phase (Q_i) together with the whole mission failure probability ($Q_{mission}$) where

$$Q_{\text{mission}} = \sum_{i=1}^{m} Q_i \qquad (1)$$

In the proposed approach, for equation (1) to be valid, the logical expression for mission failure in any phase is expressed as a combination of the causes of success of previous phases with the causes of failure for the phases being considered. In other words, the mission fails in phase *i* if the failure conditions have not been met in any of the previous *i*–1 phases and the failure conditions for phase *i* are met. Logical expressions for all failure conditions are obtained from mission phase fault trees.

The authors use BDD analysis in order to evaluate mission failure probabilities in any phase. The methodology includes the following steps. At the beginning, a fault tree for each phase is converted into a BDD using its own variable ordering scheme. The next step involves assigning time intervals over which each variable contributes to phase failure for each BDD representing the logical expression for the failure conditions being met in phase *i*. Then the logical expression for mission failure in phase *i* is built. Each logical expression is built by using the earlier mentioned combinations of causes of success for previous phases with the causes of failure for the phase being considered. The resulting BDDs are then constructed. Before the quantification process takes place the simplification process for every BDD is performed. It involves the simplification of the failure logic for each possible path from a BDD root vertex to its terminal 1 vertices which represent system failure states. Any BDD path terminating in a 1 vertex gives a cut set of the corresponding fault tree.

It is known that a BDD approach used for system failure probability quantification is more efficient than FTA. However, even BDD-based methods can be computationally intensive and require much more time when more complicated systems are analysed. Modularisation of phase fault trees was introduced in order to improve the performance of the BDD approach. This modularisation method is performed before the process of evaluation of phased failure probabilities starts, while its results are used when quantifying phased mission failure probabilities. During the modularisation process, independent subtrees are identified. In a phased mission analysis, a subtree is considered independent if it contains no basic events that appear in the rest of the fault tree of the current phase and the other phases. The rules to identify the modules (independent subtrees) are described in detail in Remenyte-Prescott et al. (2008). In the presented optimisation approach, the resulting BDDs of the modules found in the mission fault trees are incorporated when constructing the BDD for each phase in order to perform the quantitative analysis. In this case, the probability of each module failure is calculated just once and re-used as many times as a particular module appears in the mission fault trees. The application of the modularisation approach in quantification of mission failure probabilities is presented in detail by Remenyte-Prescott et al. (2008).

2.2 Mathematical optimisation problem presentation

The proposed algorithm is applied to solve a phased mission system design optimisation problem. The objective of the problem is to make alterations in the original system design in order to develop a new design system structure that would minimise the system mission failure probability. While constructing a new system design, utilisation of available resources can not exceed predefined limits and should be optimal. Thus, phased mission system design optimisation becomes a mission failure minimisation problem. Therefore, the analysed problem is presented as a single objective constrained minimisation problem:

$$\min \mathrm{Q}(\mathbf{X})_{\text{mission}}, \tag{2}$$

where **X** (*n* –dimensional vector of independent variables) is the result of the union of vectors of the system component failure probability values, i.e.:

$$\mathbf{X} = \bigcup_{i=1}^{m} \mathbf{X}_i . \tag{3}$$

Here, m is the number of phases in the mission and each $\mathbf{X}_i$ vector represents the failure probability values of the system components that appear in any minimal cut of phase i ($i = 1, 2, \ldots, m$). In other words, **X** is a vector of system components that appear in any failure event. All system components are considered as non-repairable components.

Possible alterations in the system design can be restricted due to predefined limitations to available resources and system structure itself, which includes system weight and/or volume. Therefore, in the algorithm it is considered that the system failure probability is subject to a number of constraints. The introduced constraints are grouped in two categories. The first constraint group represents the limits of cost ($Cost_{mission}$), weight ($Weight_{mission}$) and volume ($Volume_{mission}$) where cost is calculated for the whole mission time period as shown in equation (4). To use the resources efficiently, it may be useful to have minimum constraints. If only maximum limit values are needed then the minimum constraint values become equal to zero.

$$\begin{aligned} &Cost_{\min} < Cost_{mission} < Cost_{\max}, \\ &Weight_{\min} < Weight_{mission} < Weight_{\max}, \\ &Volume_{\min} < Volume_{mission} < Volume_{\max}, \end{aligned} \tag{4}$$

The second group of constraints represents the system failure probability during each phase in equation (5). Implementing these constraints allows component combinations to be identified which minimise the failure probability of the whole mission without exceeding set limits for system failure probability values during each phase.

$$\begin{aligned} &Q_1(\mathbf{X}_1) \le Q_1(\mathbf{X}_1)_{\max} \\ &Q_2(\mathbf{X}_2) \le Q_2(\mathbf{X}_2)_{\max} \\ &\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ &Q_m(\mathbf{X}_m) \le Q_m(\mathbf{X}_m)_{\max} \end{aligned} , \tag{5}$$

where, $Q_i(\mathbf{X}_i)$ identifies the ith phase failure probability, $Q_i(\mathbf{X}_i)_{\max}$ is the maximum allowed system failure probability value at phase i and m defines the number of phases in the analysed mission.

The first group of constraints [equation (4)] applies to the whole system while each constraint in the second group is associated with a single phase in the mission.

2.3 Genetic algorithm

A Genetic Algorithm (GA) was chosen as the optimisation technique to solve the phased mission design optimisation problem. The choice of GA can be attributed to one major factor. The objective function [equation (2)] does not have an explicit form and this limits the choice of optimisation technique.

A GA performance is based on the operation of populations of chromosomes, where a single chromosome represents a set of values of independent variables. The data coded in a chromosome is used to calculate values of an objective function for an analysed problem. However, a methodology employed for the evaluation of the objective function values is independent, i.e. irrelative to the optimisation technique process and does not influence the minimisation process of mission failure. Therefore, the evaluation of an objective function can be implemented according to an individually analysed problem. For example, in the proposed optimisation approach fault tree and BDD analysis were employed to find the phased mission failure probability. It means that a structure of a chromosome and its content is a 'link' between a solved problem and an optimisation process. Thus the issue of chromosome structure needs to be discussed in detail.

In the general case, independent variables of an objective function represent genes in a chromosome when using a GA for optimisation. The values of independent variables, i.e. the genes, need to be determined in order to find an optimal solution. In the proposed case, in equation (2), system components are the independent variables of the objective function. However, values of independent variables (component failure probabilities) are determined *a priori*. Secondly, the contents of the employed component sets and the number of components in the sets vary during the optimisation process. The dimension n of vector $\mathbf{X}$ [number of system components equations (2) and (3)] is not fixed and may not remain the same throughout the whole optimisation process. Since fixed length chromosomes were chosen to be used in the algorithm, it follows that neither system components, nor their failure probabilities can be used to form chromosomes.

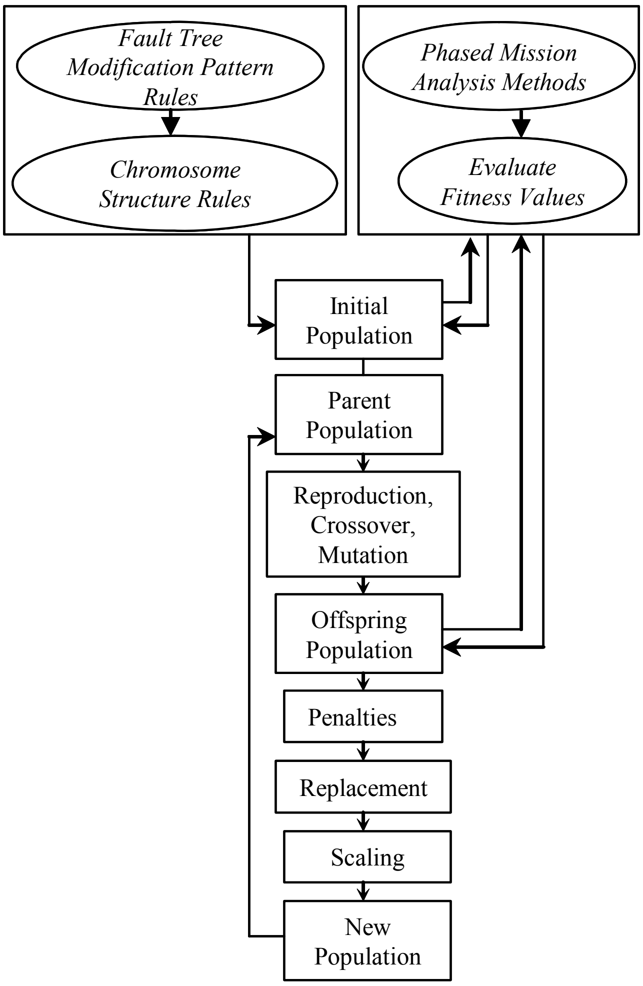
The variation in the number of system components is related to the use of design variables. A number of design variables is fixed for an analysed problem, but their values vary since an optimal set system design needs to be defined. That means the design variable values will change the contents of component set $\mathbf{X}$ and, therefore, it will result in a different objective function value [equation (2)]. Thus the optimal system design can be identified only by using different combinations of values of the design variables. It suggests that these design variables can form a chromosome structure in the GA and their values would represent gene values. Since the number of design variables remains the same throughout the whole optimisation process the chromosome length would also be fixed for an analysed problem.

The structure of a chromosome is defined as follows. In a chromosome, a certain number of binary digits, which represent a certain length gene, are allocated to store a value of every design variable in a binary format. The number of binary digits allocated to each gene is equal to the number of digits required to code the maximum possible value of an associated design variable. Thereafter, during the optimisation process each generated value of a gene is used as a value of an associated design variable by converting a binary number back to its decimal equivalent.

In the introduced approach, an analysed problem is incorporated in the optimisation process only by coding problem design variables as genes in chromosomes. The core part of the GA remains problem independent, consequently, reproduction, crossover and mutation operators are implemented irrespective of the analysed problem. The reproduction operator was implemented employing a biased roulette wheel. Each slot in the wheel is weighted in proportion to a fitness value of each population chromosome. When chromosomes in a population are coupled (the same chromosome can appear in several couples) they are crossed over employing a one-point crossover operator. During the crossover process, a bit-by-bit mutation was also carried out. Reproduction was implemented employing an algorithm described by Chambers (2001). The idea of this algorithm is to replace a parent population with an offspring population. If the best parent chromosome is fitter than the best offspring chromosome than it replaces the worst offspring chromosome.

The optimisation algorithm is summarised by the flowchart in Figure 1. It also includes penalty and scaling procedures discussed in Sections 2.4 and 2.5.

Figure 1 Optimisation algorithm flowchart



2.4 Handling of constraints

The introduced algorithm can be applied for phased mission optimisation where system design improvement is restricted by the availability of resources. In this case, the issue of possible violations of problem constraints occurs. One of the approaches used for constrained optimisation problems is that of penalty methods. The main idea of this methodology is to apply some type of penalty to solutions which violate any constraint. A scale of penalty is evaluated using a certain penalty function. In the proposed algorithm, a general adaptive penalty technique was implemented and the penalty function proposed by Coit et al. (1996) was employed:

$$\mathrm{F_p}(\mathbf{x}) = \left(\mathrm{F_{all}} - \mathrm{F_{feas}}\right)\sum_{i=1}^{nc}\left(\frac{\mathrm{d_i}\left(\mathbf{x},\mathrm{B}\right)}{\mathrm{NFT_i}}\right)^{\kappa_i} \tag{6}$$

Here, F_{all} is the best unpenalised value of the objective function yet found, F_{feas} is the best feasible value of the objective function yet found, NFT_i denotes the near-feasibility threshold that corresponds to a given constraint i, d_i ($\mathbf{x}$, B) is the magnitude of the violation of a given constraint i for solution $\mathbf{x}$, κ_i denotes a user-specified severity parameter and nc is the total number of constraints set for the problem.

In the implemented algorithm the near-feasibility threshold was defined employing a formula which allows the penalty value to be adjusted according to the search history:

$$NFT_i = \frac{NFT_{oi}}{1 + 0.1 \cdot g}\,. \tag{7}$$

NFT_{oi} represents the actual value of a constraint i and g denotes the generation number. Parameter κ_i was set to 2 in order to implement Euclidian distances between any infeasible solutions to the feasible region over all constraints.

2.5 Optimisation improvement

Quantification of a phased mission failure probability is a computationally intensive process. Therefore, it is not expedient to operate with large populations of chromosomes when using the GA for optimisation. On the other hand, employment of small populations can reduce the efficiency of the optimisation. Consequently fitness scaling was introduced in the algorithm, since it helps to improve performance of an algorithm when small populations are employed.

A linear scaling procedure proposed by Goldberg (1989) was implemented in the introduced approach. Parameters used in the linear scaling procedure are problem-independent. They depend on a population life and are found for a population in each generation.

The linear scaling method defines a linear relationship between an initial fitness value and the fitness value after the scaling:

$$\mathrm{f_{scaled}} = \mathrm{a}\mathrm{f_{initial}} + \mathrm{b}\,. \tag{8}$$

Here, $f_{initial}$ is an actual chromosomes' fitness, f_{scaled} is the chromosomes' fitness after scaling and parameters a and b are linear function coefficients. In the implemented method, these coefficients are selected so that the average fitness before scaling and the average scaled fitness are equal.

3 Application example and results

A military ship mission 'Harbour/Sea Training' was chosen as an application example of the proposed optimisation algorithm. The objective of the application is to validate the capability of the algorithm to identify a set of optimal design variable values. The application of the design variables with their identified values must minimise mission failure probability for the whole mission. Achieved ship failure probabilities for each phase must not exceed predefined limits.

The ship contains six different systems: a propulsion and power system, an electrical distribution system, a cooling water system, a hydraulic system, a hydroplane and steering system and a rudder control system. All vessel systems are analysed as non-repairable systems and independent from each other. The ship mission is divided into four consecutive phases carried out in the following order: harbour shore support, transit shallow water, receive broadcast and harbour shore support. During the first and the last phases only the electrical distribution system is in use. The fault tree for the phase includes 9 gates and 26 basic events. During the second and third phases, all vessel subsystems are in operation. Therefore, the fault trees for those phases are identical in size and comprise 26 gates and 80 events.

All six vessel subsystems are independent, i.e. any two subsystems have no common components. On the other hand, each subsystem is in use during at least two phases of the mission. Thus, each subsystem fault tree appears in a fault tree of more than one phase. It suggests that application of the fault tree modularisation methodology described in Section 2.1 can be relevant for the analysed mission. When using this approach independent modules are identified in mission fault trees. Then each module BDD is constructed and re-used to calculate the failure probability of the module in different time intervals, i.e. different phases. Since modules are smaller in comparison to the initial phase fault trees, construction of their BDDs and the quantification process using these BDDs is not complicated. On the other hand, after modularisation the resulting phase BDDs are also reduced in size. Therefore, application of the modules increases the efficiency of mission analysis. For example, 29 modules were identified when performing the quantification of the mission failure probability for the original design vessel and the efficiency of evaluation of the failure probability for the mission of different designs was evident. In addition, it also improved performance of the whole optimisation process, since mission failure probability needs to be evaluated for every set of design variable values which are decoded from the generated chromosomes. The time consumption for the optimisation process was reduced distinctively in comparison with the time required for the optimisation process when modularisation was not employed.

The data for the optimisation process of the military vessel design included fault trees for each phase, basic event failure probability values and design variables with their possible values. Constraints neither for available resources nor for failure probability values were set. Therefore, limits for constraints of the military ship failure probabilities at each phase were defined after quantitative analysis of the phased mission for the initial system design was performed. The failure probability of the first phase for the initial ship design was 1.573×10^{-3}. The second phase failure probability was 2.053×10^{-2}, the third phase failure probability was equal to 1.096×10^{-2} and the fourth phase failure probability value was 1.526×10^{-3}. The probability that the initial design military ship would fail to complete the mission was 3.459×10^{-2}. According to the obtained results the first phase failure probability limit was set to 1.573×10^{-3}, the second limit was set to 2.054×10^{-2} and correspondingly the remaining limits were set to 1.097×10^{-2} and 1.564×10^{-3}.

The adaptation of the initial design vessel phase failure probabilities as constraint values ensures that achieved improvement of the whole mission is not a result of distinguished improvement in one phase and decline of reliability in another. It provides even improvement throughout every phase and the whole mission.

Six components were chosen to be replaced in order to introduce modifications into the initial design of vessel structure. In some cases, for example, when looking for possible substitutions of two hydraulics plants, a component with different performance characteristic, i.e. different type components, was introduced for each plant. While in other cases, components were chosen to be replaced with a different number of components. The possibility to choose different types of the components was also implemented. The full list of the design variables and their values selected to characterise possible changes in the ship performance is provided in Table 1.

Table 1 List of design variables

Component	*Design variable description*	*Design variable value*
CW Pump[1]	Number of CW Pumps	3, 2, 1
	Number of CW Pumps required to trip	3, 2, 1
	Type of a CW pump	Type 1, Type 2
Feed Pump[1]	Number of Feed Pumps	4, 3, 2, 1
	Number of Feed Pumps required to Trip	4, 3, 2, 1
	Type of a Feed Pump	Type1, Type 2
Ahead Valve[1]	Number of ahead Valves	3, 2, 1
	Type of an ahead Valve	Type 2, Type 1
MG VFR[2]	Number of MG VFRs	2, 1
	Type of a MG VFR	Type 1, Type 2
External Hydraulic Plant[3]	Type of an External Hydraulic Plant	Type 1, Type 2
Main Hydraulic Plant[3]	Type of a Main Hydraulic Plant	Type 1, Type 2

Notes: [1] Propulsion & Power System, [2] Electrical Distribution System, [3] Hydraulic System.

In Section 2.1, it was mentioned that before the optimisation process starts, phase fault trees are reconstructed in order to associate them with different possible values of design variables. These fault trees include groups of house events and new basic events linked with all possible values of design variables chosen for system improvement. An example of the main hydraulics subsystem fault trees is presented in Figure 2 and Figure 3.

The fault tree for the hydraulics system of the original design ship is shown in Figure 2 while Figure 3 presents how the subsystem fault tree has been modified due to the implementation of a choice of a type of a main hydraulics plant.

Values of genetic operators, i.e. crossover and mutation rates, as well as population size influence the performance of a GA. However, the exact rules for choosing values of GA parameters do not exist. In the military ship mission optimisation case, these parameter values were chosen according to problem characteristics and testing results. For example, in the analysed case a size of population is restricted due to the computationally intensive process of mission failure quantification. Therefore, it was decided to restrict the size of the populations to 30 chromosomes. Then genetic parameter values were defined by running algorithm simulations and using different combinations of parameter values. Mutation rates were used equal to 0.001, 0.005 and 0.01 and crossover rate values were equal to 0.75, 0.8 and 0.95.

Figure 2 A fault tree for the original design hydraulics subsystem

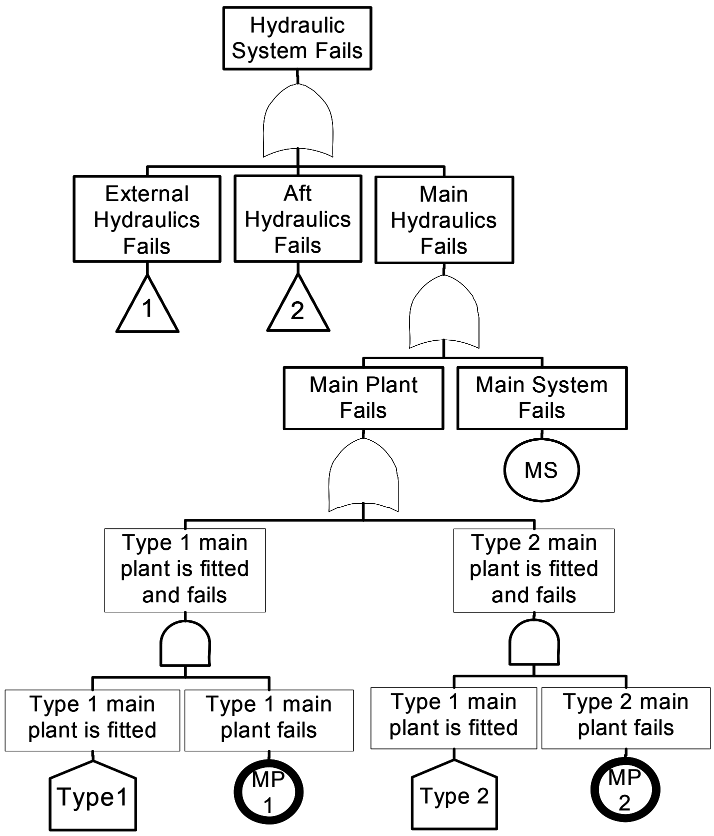
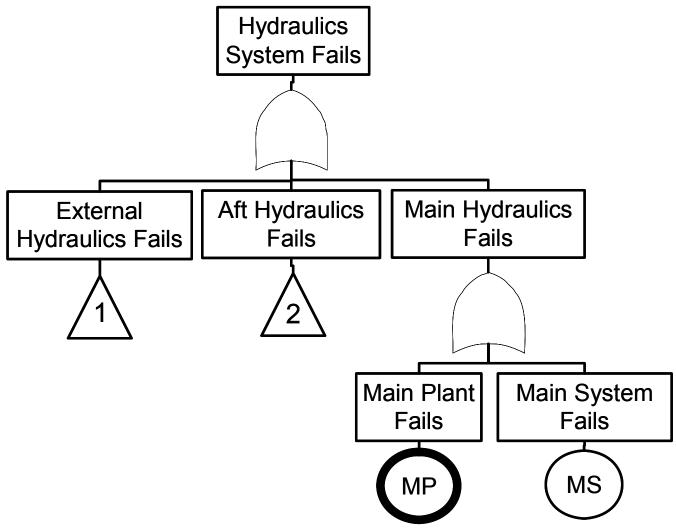


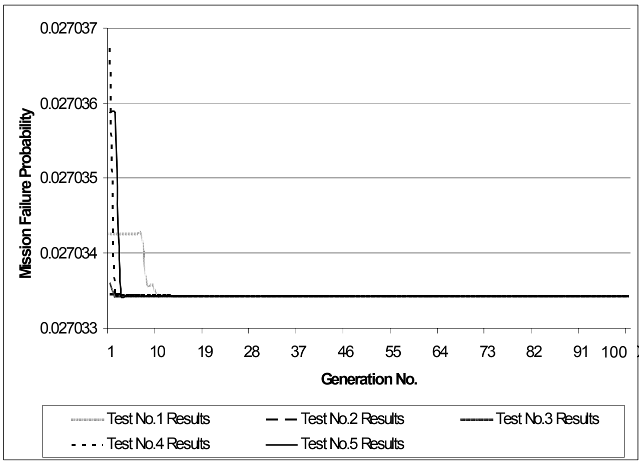
Figure 3 A fault tree for the modified hydraulics subsystem



Optimisation simulations were run five times for each combination of GA parameters. Following this, average numbers of generations required to find the minimal failure probability values were derived. These numerical values were used as a comparison measure to evaluate the performance of the optimisation algorithm when using different GA parameter values. Even though the results of all simulations converged to the same mission failure probability value, the smallest average number of generations required to find the minimal failure probability value was obtained when using a 30 chromosome population, a crossover rate equal to 0.75 and a probability of mutation equal to 0.001. This set of GA parameter values was used to perform the optimisation of the military vessel mission for visual representation of the results. The optimisation simulation was carried out also five times in order to observe the tendency in convergence of the results. Each time the process was terminated after 100 generations.

Results presented in Figure 4 are the average mission failure probability values for each generation. They show that the dispersion of results for each run is rather small and that each time the objective function values converge to the minimal failure probability value. The optimal failure probability values for each generation shown in Figure 5 confirm convergence of results to the global minimum for the problem. The convergence of results is relatively rapid since the optimal failure probability value appears in the first 20 generations in each simulation case.

Figure 4 Average mission failure probability values of each generation for five runs



The minimal military ship mission failure probability obtained during the optimisation processes was equal to 2.70334343×10^{-2} while the mission failure probability of the initial design ship was 3.45948602×10^{-2}. The failure rate was minimised due to a number of component replacements that were made in constituent subsystems of the vessel. As a result an optimal vessel design was composed. The optimally designed ship now includes the number of new additional components that have replaced four out of six chosen components from Table 1. Two components, i.e. external hydraulics and main hydraulics plants were not replaced. The list of the new components is presented in Table 2.

Figure 5 Minimal mission failure probability values for each generation

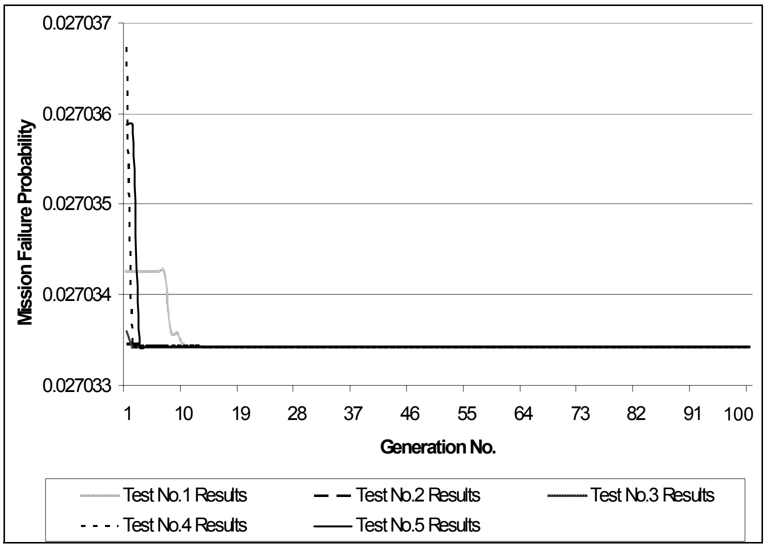


Table 2 Values of design variables for the optimal ship design

Changeable component	*Design variable description*	*Design variable value*
CW Pump[1]	Number of CW Pumps	3
	Number of CW Pumps Required to trip	1
	Type of a CW pump	Type 2
Feed Pump[1]	Number of Feed Pumps	4
	Number of Feed Pumps required to Trip	1
	Type of a Feed Pump	Type 2
Ahead Valve[1]	Number of ahead Valves	3
	Type of an ahead Valve	Type 1
MG VFR[2]	Number of MG VFRs	2
	Type of a MG VFR	Type 1

Notes: [1] Propulsion & Power System, [2] Electrical Distribution System.

4 Conclusions

The introduced algorithm is proposed for phased mission system failure probability minimisation problems. The minimisation of the failure probability is implemented by substituting a number of system components with chosen new ones. In some cases, the changes of system structure may result in increased expenses and/or maintenance down time and/or system weight and volume. Therefore, the algorithm also incorporates the possibility to set limits for these resources and the analysed system design characteristics. As a result, the algorithm determines the case where the phased mission system failure probability is minimised and the utilization of available resources is optimised.

In the developed algorithm, fault trees were used to represent system failure modes of all mission phases for each system design. The fault trees are converted to corresponding BDDs by following certain rules in order to perform quantification analysis of the phased mission system. Additionally, the fault tree modularisation is applied before conversion into BDDs is performed. The introduced fault tree modularisation significantly improved efficiency of mission failure quantification and the whole optimisation process.

A simplified four-phase military ship mission was employed as an application example. During the optimisation process a set of design variable values was identified for improvement of ship performance. Implementation of alterations in ship structure according to the design variables resulted in minimisation of mission failure probability. The results also indicates that the global minimum of the optimisation problem has been found, since the objective function values converged to one value equal to 2.70334343×10^{-2}.

Given the applicability of the method to the example mission, the next step would be to analyse more complicated phased mission systems. It is envisaged that this will introduce additional computational intensity which may incur a processing time issue. Therefore, future work will focus on improving the performance of the algorithm and subsequently its application for larger systems.

References

Alam, M. and Al-Saggaf, U.M. (1986) 'Quantitative reliability evaluation of repairable phased-mission systems using Markov approach', *IEEE Transactions on Reliability*, Vol. 35, No. 5, pp.498–503.

Chambers, L.D. (Ed.) (2001) *The Practical Handbook of Genetic Algorithms*, Chapman and Hall/CRC, Boca Raton, FL.

Coit, D.W., Smith, A.E. and Tate, D.M. (1996) 'Adaptive penalty methods for genetic optimization of constrained combinatorial problems', *INFORMS Journal on Computing*, Vol. 8, No. 2, pp.173–182.

Esary, J.D. and Ziehms, H. (1975) 'Reliability of phased missions', *Reliability and Fault-Tree Analysis*, pp.213–236.

Goldberg, D.E. (1989) *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Wokingham.

Kim, K. and Park, K.S. (1994) 'Phased-mission system reliability under Markov environment', *IEEE Transactions on Reliability*, Vol. 43, No. 2, pp.301–309.

La Band, R.A. and Andrews, J.D. (2004) 'Phased mission modelling using fault tree analysis', *Proceedings of the Institution of Mechanical Engineers, Part E: Journal of Process Mechanical Engineering*, Vol. 218, No. 2, pp.83–91.

Ma, Y. and Trivedi, K.S. (1999) 'An algorithm for reliability of phased-mission systems', *Reliability Engineering & System Safety*, Vol. 66, pp.157–170.

Pattison, R.L. (1999) *Safety System Design Optimisation*, PhD Thesis, Loughborough University, Loughborough.

Prescott, D.R., Remenyte-Prescott, R., Reed, S., Andrews, J.D. and Downes, C.G. (2008, January) 'A fast analysis method for phased missions using the binary decision diagram method', *Reliability Engineering & System Safety*.

Remenyte-Prescott, R., Andrews, J.D. and Downes, C.G. (2008, January) 'A phased mission systems reliability analysis for mission planning of autonomous vehicles', *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*.

Susova, G.M. and Petrov A.N. (1997) 'Markov model-based reliability and safety evaluation for aircraft maintenance-system optimization', *Proceedings of Annual Reliability and Maintainability Symposium*, pp.29–36.