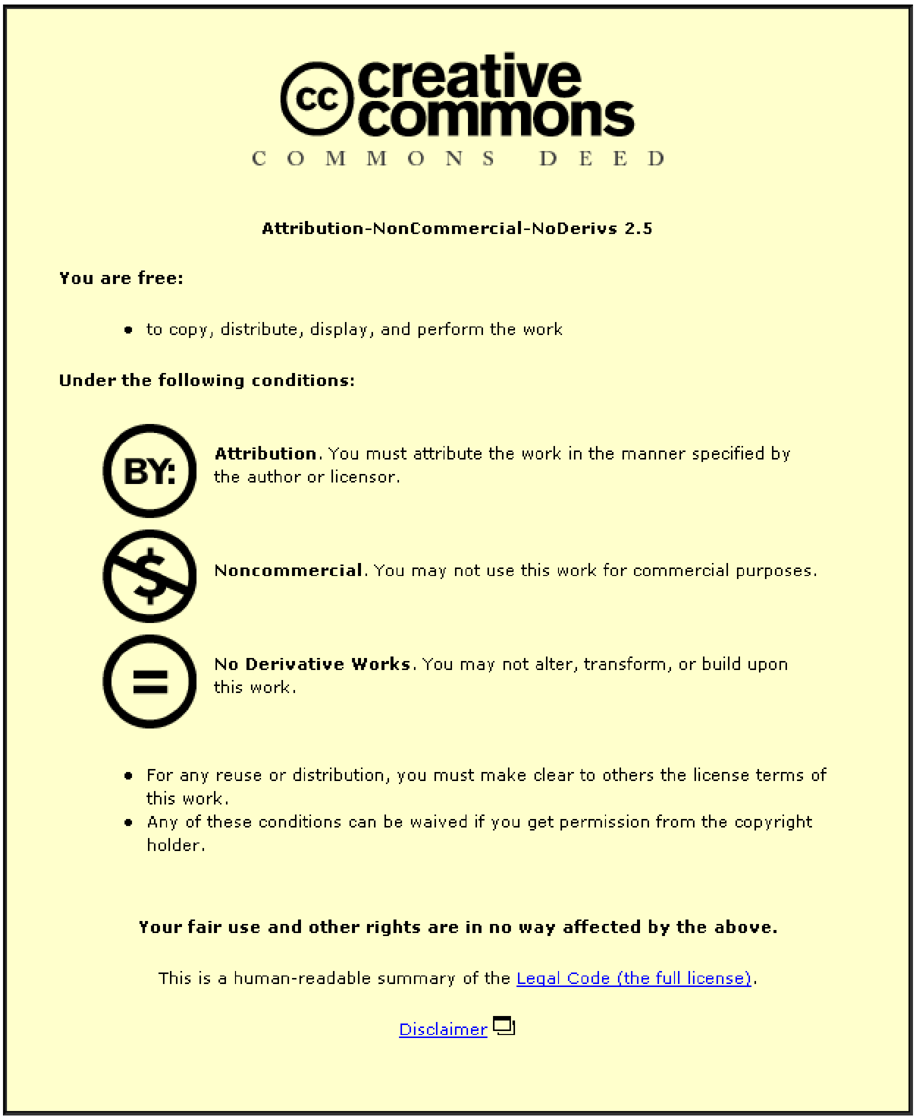

Originally published by the System Safety Society, in the Proceedings of the 26th International System Safety Conference, held in Vancouver 2008

Phased mission failure minimisation using optimal system design

D. Astapenko & L.M. Bartlett

Department of Aeronautical and Automotive Engineering, Loughborough University, Leicestershire, UK, LE11 2TU.

Contact authors: L.M.Bartlett@lboro.ac.uk / D.Astapenko@lboro.a.uk

ABSTRACT

A phased mission system represents a system whose performance is divided into consecutive non-overlapping phases. The operation of a phased mission system can be improved by either introducing better performing components or by increasing the number of redundant components. At the same time, such design alterations can influence how available resources are utilised. The focus of this paper is to develop an optimisation method to construct an optimal design case for a phased mission system, with the aim of maximising its availability and ensuring optimal usage of available resources throughout all phases. The developed method is based on an approach where an individual phase is treated as a standard single phase system. Thus, to solve the whole phased mission optimisation problem each phase design is analysed individually, whilst dependencies between different phases are also included in the analysis in order to find the failure probability value of each phase. The implemented optimisation method employs Fault Tree Analysis (FTA) to represent system performance and Binary Decision Diagrams (BDDs) to quantify each phase failure probability. A Single Objective Genetic Algorithm (SOGA) has been chosen as the optimisation technique. A simple Unmanned Aerial Vehicle (UAV) mission has been selected to demonstrate the methods application. Results of the analysis are discussed.

1 INTRODUCTION

A phased mission system represents a system where performance is analysed as a sequence of non-overlapping phases. To complete a mission the system is required to accomplish a specific task without failure in each phase. A classical example of a phased mission system is for an aircraft which undergoes three phases: take-off, cruise and landing. The aircraft completes its mission if all three tasks have been completed successfully.

Originally published by the System Safety
Society, in the Proceedings of the 26th International System Safety
Conference, held in Vancouver 2008

Generally, methods used for phased mission system analysis can be grouped in two categories: Markov analysis based approaches and combinatorial methods. Combinatorial methods include fault tree and binary decision diagram (BDD) approaches. The earliest applications of fault tree analysis in the analysis of non-repairable phased missions were made by Esary J.D. & Ziehms, H. (1975).

If a phased mission system is repairable or there are dependencies between component failures or phases, it is generally difficult to find the exact system reliability. In this case Markov reliability models are employed, as discussed by Kim & Park (1994).

Both combinatorial and Markov methods have some drawbacks. For example, in Markov models for systems with n components, up to 2^n equations are needed to represent each phase. In the combinatorial approach, however, the problem size expands with increasing number of phases. Therefore, employing a combination of both approaches for the same problem can help to overcome these individual drawbacks and enable the analysis of more complicated problems. As such, Ou Y. et al. (2002) and Wang D. & Trivedi K.S. (2007) introduced new approaches to analyse phased mission systems adopting this approach.

A large number of systems which can employ mechanical, chemical, electronic and nuclear technologies can be analysed as phased mission systems. Similarly, as such, transportation networks and indeed military missions can be analysed as phased mission systems. Thus system optimisation in terms of reliability, cost, etc is extremely relevant. In spite of its importance, however, there is limited demonstrated evidence in the literature for research that focusses on phased mission optimisation; Susova G.M. & Petrov A.N. (1997) introduced a model based on Markov homogeneous process which can be used to minimize operation cost and optimise flight safety.

The purpose of this paper is to introduce an algorithm that aims to determine the optimal system design of a phased mission system. The algorithm solves for minimum overall mission failure probability within the context of pre-defined design constraints and resources.

The fault tree analysis approach introduced by La Band R.A. & Andrews J.D. (2004) together with the BDD approach presented by Remenyte-Prescott R. (2007), which were applied for phased mission reliability analysis, have been employed. The fault tree approach enables the failure causes in each phase and the whole mission to be described, while the BDD approach is more efficient for mathematical analysis (probability evaluation). To find an optimal system design a single objective Genetic Algorithm (GA) was chosen as the optimisation technique.

2 PHASED MISSION DESIGN OPTIMISATION ALGORITHM

2.1 *Optimisation problem introduction*

A phased mission system design optimisation problem is represented as a general single objective minimisation problem. The problem is stated as a minimisation of failure probability for a system:

$$\min Q(\mathbf{X})_{mission}, \qquad (1)$$

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

$$\mathbf{X} = \bigcup_{i=1}^{m} \mathbf{X}_i \,. \qquad (2)$$

Here, *m* is the number of phases in the mission and each $\mathbf{X}_i$ vector represents system components that appear in any minimal cut set of a fault tree of phase *i (i* = 1, 2*, …, m*). In other words, **X** is a vector of system components that appear in any failure event.

For a general optimisation problem, values of vector components need to be defined. In the introduced problem (1) a particular set of system components which correspond to the minimal system unavailability value are searched for. Therefore, the contents of the employed component sets and the number of components in the sets vary through the analysis. The dimension *n* of vector **X** (number of system components) is not fixed and may not remain the same during the whole optimisation process.

In the algorithm it is considered that the system failure probability is subject to a number of constraints. The constraints can be grouped in two categories. The first constraint group represents the limits of the available resources, such as cost ($Cost_{mission}$), maintenance down time ($MDT_{mission}$), weight ($Weight_{mission}$) and volume ($Volume_{mission}$). To use resources efficiently it may be useful to have minimum constraints. If limits are set just to maximum values then minimum constraint values easily become equal to zero.

$$\begin{aligned} &Cost_{\min} < Cost_{mission} < Cost_{\max}, \\ &MDT_{\min} < MDT_{mission} < MDT_{\max}, \\ &Weight_{\min} < Weight_{mission} < Weight_{\max}, \\ &Volume_{\min} < Volume_{mission} < Volume_{\max}, \end{aligned} \qquad (3)$$

where cost and maintenance down time are calculated for an analysed time period.

In this research, the limitations on available resources are applied to the whole mission and are not analysed for each phase individually.

The second group of constraints represents the system failure probability during each phase (4). Implementing these constraints allows component combinations to be identified that minimise the failure probability of the whole mission without exceeding limits set for system failure probability values during each phase.

$$\begin{array}{l} Q_1(\mathbf{X}_1) \le Q_1(\mathbf{X}_1)_{\max} \\ Q_2(\mathbf{X}_2) \le Q_2(\mathbf{X}_2)_{\max} \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ Q_m(\mathbf{X}_m) \le Q_m(\mathbf{X}_m)_{\max} \end{array}, \qquad (4)$$

where, $Q_i(\mathbf{X}_i)$ identifies the *i*th 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.

To solve the introduced single objective optimisation problem, a Genetic Algorithm (GA) was chosen as the optimisation technique. The choice of this technique can be attributed to one major factor. The objective function (1) does not have an explicit form, which makes the choice of optimisation technique limited.

2.2 *Design variables*

System components are the independent variables of the objective function (1). In a general optimisation case, values of the independent variables need to be determined in order to find an optimal solution. In the proposed case values of independent variables (component failure probabilities) are determined a priori and therefore a set of components needs to be found which would give the optimal objective function value.

Usually, in trying to improve system performance, a certain number of components are chosen to be replaced. A different number of redundant components and / or components of different types, i.e. components that have different characteristics, can replace existing components. Therefore the notation "*design variables*" is used to 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.

Different design variable values will change the component set **X**. Thus the optimal system design can be defined just by the optimal values of design variables. It suggests that just design variables need to be coded in the chromosomes of the GA and their values need to be analysed.

To include all possible design variables requires specific forms for the fault trees of the system phases to be used. These fault trees include groups of associated house events. The house events are employed to activate certain fault tree branches according to the values of the design variables. When fault tree branches with house events set to 0 (inactive house events) are eliminated, fault trees for phases are constructed. The obtained fault trees represent a particular system design. The methodology of introducing house events associated with design variables into fault trees was discussed by Pattison R.L. (1999).

The maximum possible values of the chosen design variables are used to construct a chromosome structure used in the genetic algorithm for optimization purposes. In a chromosome a certain number of binary digits, which represent a certain length gene, are allocated to store a value of every variable in a binary format. Thereafter, each generated value of a gene is used as a value of an associated design variable.

Figure 1 illustrates how design variables and fault tree house events are linked with GA chromosomes.

2.3 *Objective function evaluation*

In general, the result of an objective function when using any vector **X** (1) is the failure probability value of a mission. A methodology introduced by La Band R.A. & Andrews J.D. (2004) was employed in order to determine the failure probability of a system. The method provides failure probabilities for each phase (Q_i) together with the whole mission failure probability ($Q_{mission}$) where

$$Q_{mission} = \sum_{i=1}^{m} Q_i \qquad (5)$$

For equation 5 to be valid, fault trees for each phase need to be modified according to certain rules.

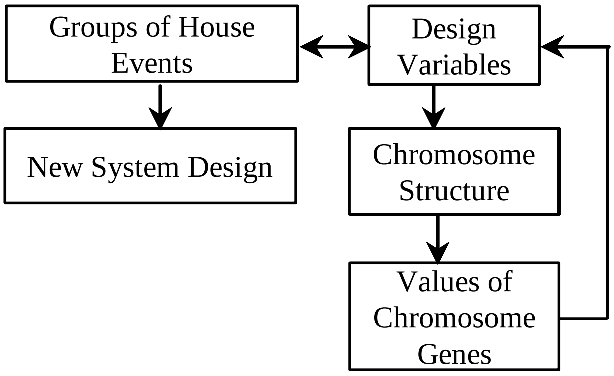

Figure 1. Links between different data groups.

When altering an *i*th phase fault tree any event of a component failure needs to be represented as an "OR" gate with a number of input events. Each input event represents a component failure event in a certain phase *j*, where *j* = 1, 2, …*i*-1 and one event is a component failure in phase *i*. This shows that if a component fails in phase *i*, it could have failed during any phase *j* up to and including phase *i*, since the component is non-repairable. As an example, the failure of component *A* in phase 3 is presented in Figure 2. Here, A_1 is the failure of component *A* in phase 1, A_2 is the failure of the component in phase 2 and A_3 represents the failure of component *A* in phase 3.

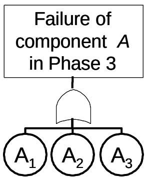

Figure 2. Representation of event *A* as an "OR" gate.

If a system fails in phase *i*, it means it could not have failed during any previous phase *j* (*j* = 1, 2, …, *i*-1). Therefore, system failure in phase *i* is represented by an "AND" gate that incorporates the success of previous phases *j* (using NOT logic) and the failure for the phase *i*. Figure 3 represents the failure of a system in phase 3. Phase 3 failure is shown as a combination of successes in phases 1 and 2 and failure in phase 3. In phase 3 components *A, B* and *C* are replaced using "OR" gates to indicate that components could have failed during phase 1, 2 or 3, as shown previously in Figure 2. The overall mission failure probability is then equal to the sum of failure probabilities for all phases (5).

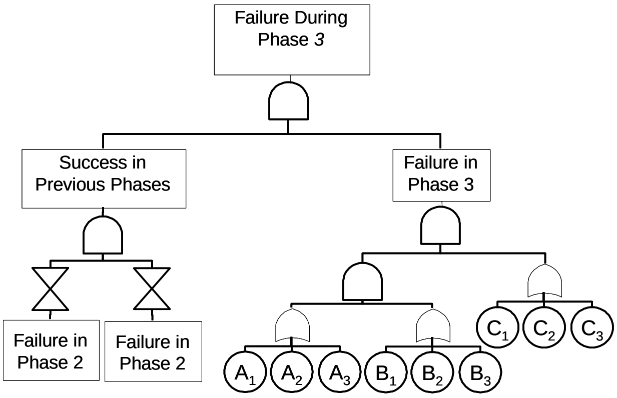

Figure 3. An example of system failure in phase 3.

Hereafter, phased mission analysis using fault trees can become too complicated when the fault trees are large. Thus to evaluate mission failure probability, fault trees for each phase are converted to BDDs. A methodology detailed by Remenyte-Prescott R. (2007) was correspondingly employed. At first, each phase fault tree is converted to a BDD. Next, the created BDDs are altered using the earlier described methodology introduced by La Band R.A. & Andrews J.D. (2004). The obtained BDDs are employed to evaluate particular design system failure probabilities for phases and the sum of their values determines the failure probability value for the whole mission.

2.4 *Genetic algorithm*

As mentioned earlier, a single objective GA was employed as the optimisation technique in the proposed algorithm for the phased mission design optimisation problem. Additionally, fault tree and BDD analysis were employed to find objective function values. The core part of the genetic algorithm remains simple using reproduction, crossover and mutation operators.

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 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 L. (ed.) (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 4. It also includes scaling and penalty procedures discussed in the following section.

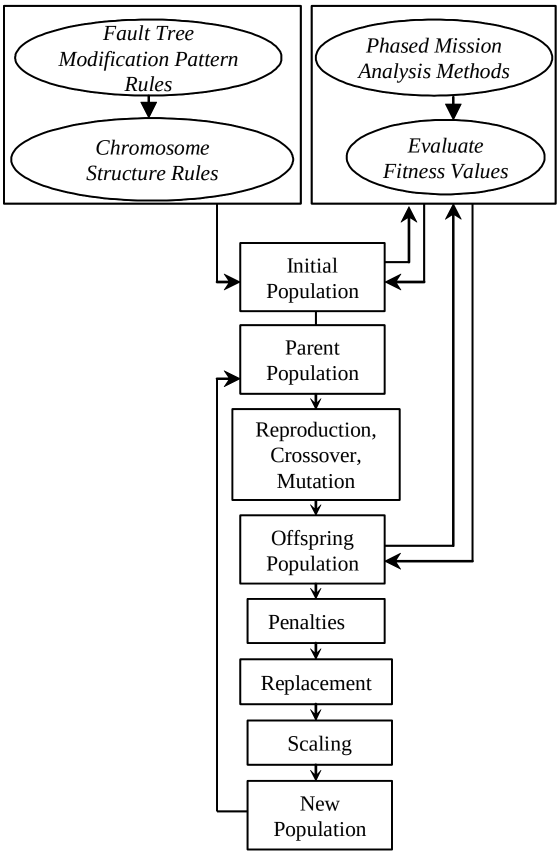
Fault Tree Modification Pattern Rules
Chromosome Structure Rules
Phased Mission Analysis Methods
Evaluate Fitness Values
Initial Population
Parent Population
Reproduction, Crossover, Mutation
Offspring Population
Penalties
Replacement
Scaling
New Population

Figure 4. Optimisation algorithm flowchart.

2.5 *Scaling and Constraints*

A penalty application was chosen as an approach to deal with possible problem constraints. The main idea of this methodology is to apply some type of penalty to solutions which violate any constraint. A penalty function proposed by Coit D.W. et. al. (1996) was employed in the algorithm:

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

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 **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:

$$NFT_i = \frac{NFT_{oi}}{1 + 0.1 \cdot g}. \qquad (7)$$

Here 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.

In order to improve the performance of the algorithm, fitness scaling was introduced. Fitness scaling is especially valuable when small population genetic algorithms are employed.

A linear scaling procedure proposed by Goldberg D.E. (1989) was introduced in the algorithm. 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:

$$f_{scaled} = af_{initial} + b. \qquad (8)$$

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

The proposed optimisation algorithm was applied to analyse a simplified unmanned aerial vehicle (UAV) mission. The aim of the optimisation was to find the optimal UAV design that would improve availability (minimise unavailability) for the whole mission and at the same time achieve vehicle failure probabilities for each phase that do not exceed pre-defined limits.

The UAV mission comprises of six phases carried out in the following order: take-off, climb, en-route in controlled airspace, en-route in uncontrolled airspace, descent and land. There are 28 different basic events used in mission fault trees which can be grouped into two categories. Basic events that represent failures of UAV components are called internal basic events. Basic events which are associated with external factors causing mission failure belong to a group of external basic events. These events are such as bird strike, air traffic control failure, communication error and storm.

To perform the optimisation analysis five UAV components were chosen to be replaced. The list of design variables selected to characterise improvements in UAV performance is provided in Table 1.

Table 1. List of design variables.

Changeable component	Description of modifications	Possible values of design variables
Landing gear	Type of a landing gear	*type1, type 2*
Antiskid valve	Number of feed antiskid valves	*2, 1*
Cross feed valve	Number of cross feed valves	*3, 2, 1*
	Type of a cross feed valve	*type1, type 2*
Pump	Number of pumps	*2, 1*
	Type of a pump	*type1, type 2*
	Minimal number of pump failures causing system failure	*2, 1*
Navigation system	Number of sub navigation systems	*2, 1*
	Minimal number of navigation system failures causing system failure	*2, 1*

In Section 2.2 it was mentioned that the implementation of design variables requires a special form of phase fault trees. These fault trees include groups of house events associated with design variables chosen for system improvement. An example of the first phase fault tree (*take-off*) is presented in Figure 5. It shows how the phase fault tree is modified due to the implementation of the choice of a type of a landing gear.

Before any design changes have been implemented, performance of the original, i.e. initial design UAV, was analysed in order to set limits for system failure probabilities at each phase. The failure probability of the first phase for the initial design UAV was 0.0306. The second phase failure probability was 0.0174, the third phase failure probability was equal to 0.0077, the fourth phase failure probability value was 0.0058, the fifth phase failure probability increased up to 0.0301 and after the sixth increased to 0.0435. The probability that the initial design UAV would fail to complete the mission was 0.1359. Phase failure constraint data was chosen according to these phase failure probability values to ensure that the unreliability of each phase of a new design UAV would not increase. The first phase failure probability limit

Originally published by the System Safety
Society, in the Proceedings of the 26th International System Safety
Conference, held in Vancouver 2008

was set to 0.04, the second limit was set to 0.02 and correspondingly the remaining limits were set to 0.008, 0.006, 0.04 and 0.05.

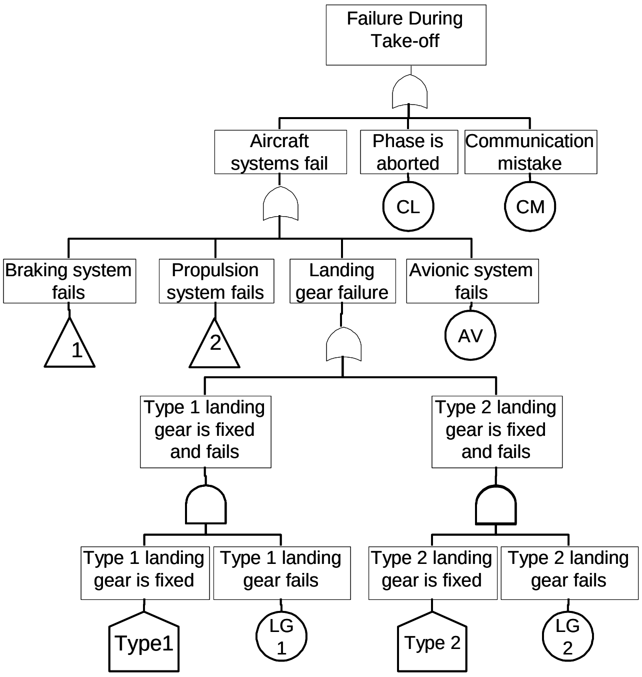

Figure 5. Fault tree for phase *take-off*.

When applying a genetic algorithm its parameter values influence the performance of the algorithm. Thus several tests were carried out to find such parameter combination which could be used when carrying out UAV design optimisations. Different combinations of population size, crossover rate and mutation rate values were chosen. Three population sizes were analysed: 50, 30 and 10 chromosomes. Population size had the biggest influence on optimisation duration. Mutation rates were chosen equal to 0.001, 0.005 and 0.01 and crossover rate values were equal to 0.75, 0.8 and 0.95.

The number of phases defines the scale of an analysed system and therefore the time required to find failure probability values for each phase, but it does not influence the performance of the optimisation algorithm in terms of the results convergence. As such, the right combination of GA parameters were progressively determined using two phases in the mission in order to avoid the much greater time consumption required for a six-phase mission. The best result, i.e. the smallest average number of generations required to find the minimal failure probability values was obtained when using a 30 chromosome population, a crossover rate equal to 0.75 and a probability of mutation equal to 0.001. These parameter values were used to perform the optimisation of the actual six-phase UAV mission. The optimisation simulation was carried out five times in order to show consistency in the convergence of results. Each time the process was terminated after 100 generations.

Results presented in Figure 6 are the average mission failure probability values for each generation. They show that the objective function values converge to average minimal failure probability value. The optimal failure probability values for each generation shown in Figure 7 confirm convergence of results on the global minimum for the problem.

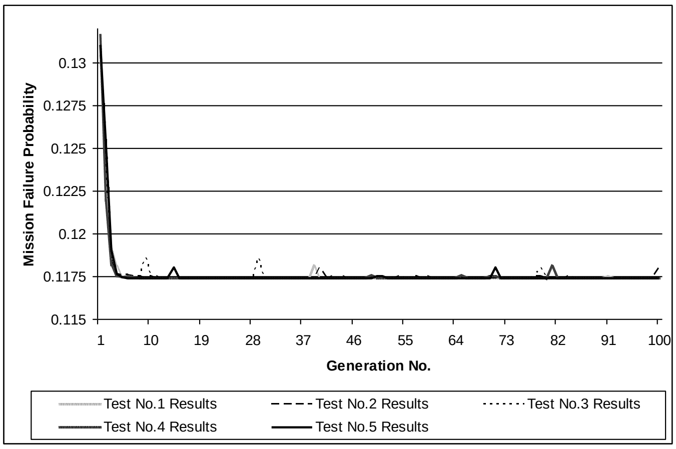

Figure 6. Average mission failure probability values of each generation for five runs.

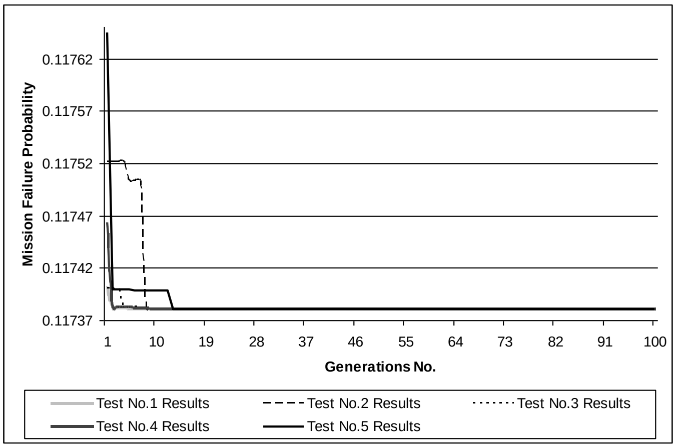

Figure 7. Minimal mission failure probability values for each generation.

The minimal mission failure probability obtained from the model was equal to 0.117380741, representing the failure probability for the optimal UAV design, while the mission failure probability of the initial design UAV was 0.1358712415. The optimally designed vehicle now includes new components that have replaced the some of the original components. The list of the new components is presented in Table 2.

Table 2. Values of design variables for the optimal system design. (q=0.117380741)

Changeable component	Design variable description	Design variable value
Landing gear	Type of a landing gear	*type1*
Antiskid valve	Number of feed antiskid valves	*2*
Cross feed valve	Number of cross feed valves	*3*
	Type of cross feed valves	*type 2*
Pump	Number of pumps	*2*
	Type of a pump	*type 2*
	Minimal number of pump failures causing system failure	*2*
Navigation system	Number of sub navigation systems	*2*
	Minimal number of navigation system failures causing system failure	*2*

4 CONCLUSIONS

The introduced algorithm is employed to minimise phased mission system failure. The failure is minimised by defining a set of system components that constitutes an optimal system design and contributes to the improvement of system performance.

In the presented algorithm fault tree analysis was used to represent system failure modes during the whole mission for each system design. Fault tree and BDD approaches were also employed to evaluate the failure probabilities for each phase and the whole mission for different system designs. A single objective GA was chosen as an optimisation technique.

The developed algorithm has been successfully applied to find an optimal UAV phased mission design which minimises the whole mission failure probability. Obtained optimisation results that represent mission failure probability values converged to the global minimum. A set of UAV components characterising the optimal vehicle design was identified. As a result, the optimal design UAV mission failure probability represented the global minimum of the optimisation process.

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

REFERENCES

Chambers L.D. (ed.) 2001. *The practical handbook of genetic algorithms.* Boca Raton, Fla: Chapman & Hall/CRC.

Coit D.W., Smith A.E. & Tate D. M. 1996. Adaptive penalty methods for genetic optimization of constrained combinatorial problems. *INFORMS Journal on Computing* 8 (2): 173-182

Esary, J.D. & Ziehms, H.1975. Reliability of phased missions. *Reliability and Fault-Tree Analysis*: 213-236

Goldberg D.E. 1989. *Genetic algorithms in search, optimization, and machine learning*. Wokingham: Addison-Wesley.

Kim K. & Park K. S. 1994. Phased-mission system reliability under Markov environment. *IEEE Transactions on reliability* 43(2): 301-309.

La Band R.A. & Andrews J.D. 2004 Phased mission modelling using fault tree analysis. *Proc. Instn Mech. Engrs, Part E: Journal of Process Mechanical Engineering* 218(2): 83-91.

Ou P.,Meshkat L. & Dugan J. B. 2002. Multi-phase reliability analysis for dynamic and static phases. *Proceedings of annual reliability and maintainability symposium*: 404-410.

Pattison R.L. 1999. *Safety system design optimisation (PhD Thesis)*. Loughborough: Loughborough University.

Remenyte-Prescott R. 2007. *System failure modelling using binary decision diagrams (PhD Thesis)*. Loughborough: Loughborough University.

Susova G.M & Petrov A. N. 1997. Markov model-based reliability and safety evaluation for aircraft maintenance-system optimization. *Proceedings of annual reliability and maintainability symposium*: 29-36.

Wang D & Trivedi K.S. 2007. Reliability analysis of phased-mission system with independent component repairs. *IEEE Transactions on reliability* 56(3): 540-551.