An Exposition on Solving the Joint Level of Repair Analysis-Spares Problem using a Multi-Objective Genetic Algorithm

Raman Pall
CJOC OS Operational Research and Analysis Team

Slawomir Wesolkowski
Director Air Staff Operational Analysis

Matthew Dozois
University of Waterloo Co-op student

DRDC CORA TM 2013–159
September 2013

Defence R&D Canada
Centre for Operational Research and Analysis

CJOC Operational Research and Analysis
Canadian Joint Operations Command

An Exposition on Solving the Joint Level of Repair Analysis-Spares Problem using a Multi-Objective Genetic Algorithm

Raman Pall
CJOC OS Operational Research and Analysis Team

Slawomir Wesolkowski
Director Air Staff Operational Analysis

Matthew Dozois
University of Waterloo Co-op student

Defence R&D Canada – CORA
Technical Memorandum
DRDC CORA TM 2013–159
September 2013

Principal Author

Original signed by Raman Pall

Raman Pall

Approved by

Original signed by Isabelle Julien

Isabelle Julien
Section Head (Land and Operational Command)

Approved for release by

Original signed by Paul Comeau

Paul Comeau
Chief Scientist

Abstract

Level of repair analysis (LORA) is often defined as the problem of determining whether a component should be repaired or discarded upon its failure, and the location in the repair network to do such work. A related problem is the determination of the optimal number of spare components for a given piece of equipment. The most common approaches in the literature on developing a possible spare provisioning decision model are simulation and mathematical programming. Although these two problems (LORA and spare provisioning) are interdependent, they are seldom solved simultaneously due to the complicating nature of the relationships between spare levels and system availability.

The need to address LORA and the sparing problems simultaneously has attracted increased attention from the Department of National Defence (DND). In this technical memorandum, the use of a multi-objective genetic algorithm (specifically the Non-dominated Sorting Genetic Algorithm II) is proposed to solve this problem, with optimization objectives as minimizing repair costs (*e.g.*, spare parts, spares transportation, spares storage) and maximizing operational availability. The approach uses a Monte Carlo simulation to generate scenarios based on a dataset which includes failures of the components and their associated times of failure. The objective functions are computed at each genetic algorithm generation based on all generated scenarios. Examples are given on a realistic dataset in order to illustrate the type of trade-off analyses that can be carried out. Finally, further work on a large real (not just realistic) dataset would need to be carried out for DND to benefit fully from the research.

Résumé

L'analyse du niveau de réparation (ANR) est souvent définie comme le processus qui consiste à déterminer si un composant défectueux doit être réparé ou jeté, et à quel endroit dans le réseau la réparation doit se faire le cas échéant. On doit aussi déterminer le nombre optimal de composants de rechange que l'on peut garder en inventaire pour une pièce d'équipement donnée. La simulation et la programmation mathématique sont les méthodes qui reviennent le plus souvent dans la documentation portant sur l'élaboration d'un modèle décisionnel d'approvisionnement en pièces de rechange. Bien que ces deux problèmes (l'ANR et le stockage des pièces de rechange) soient interdépendants, ils sont rarement résolus simultanément en raison de la complexité des relations entre les niveaux de pièces de rechange et la disponibilité du système.

Le ministère de la Défense nationale (MDN) porte une attention de plus en plus grande à la nécessité de s'attaquer simultanément au problème de l'ANR et à celui de l'approvisionnement en pièces de rechange. Dans ce document technique, nous proposons de résoudre ce problème au moyen d'un algorithme génétique à objectifs multiples (le *Non-Dominated Sorting Genetic Algorithm II*), les objectifs d'optimisation étant de réduire le plus possible les coûts liés à la réparation (p. ex. pièces de rechange, transport des pièces, stockage des pièces) et de maximaliser la disponibilité opérationnelle. Cet algorithme utilise une simulation de Monte Carlo pour créer des scénarios à partir d'un ensemble de données faisant état des pannes des composants et du moment de la panne. Les fonctions objectifs sont calculées à chaque exécution de l'algorithme sur la base de tous les scénarios générés. Nous présentons des exemples fondés sur un ensemble de données réaliste pour illustrer le genre d’analyses d’arbitrage qui peuvent être faites. Enfin, pour que le MDN tire pleinement profit de ces recherches, il serait normal de pousser plus loin l’analyse en s’intéressant cette fois à un grand ensemble de données réel (et non seulement réaliste).

Executive summary

An Exposition on Solving the Joint Level of Repair Analysis-Spares Problem using a Multi-Objective Genetic Algorithm

Raman Pall, Slawomir Wesolkowski, Matthew Dozois; DRDC CORA TM 2013–159; Defence R&D Canada – CORA; September 2013.

Background: Level of repair analysis (LORA) is often defined as the problem of determining whether a component should be repaired or discarded upon its failure, and the location in the repair network to do such work. A related problem is the determination of the optimal number of spare components for a given piece of equipment.

The complexity of LORA lies in the fact that most of the complex equipment has modular design, and maintenance is carried out by exchanging parts at locations at different hierarchy levels. Upon failure of a system, the component which is determined to be responsible for the overall failure of the system is identified. Occasionally, this component can be broken down further into smaller subsystems, and the source of the failure can be traced to a single part. The types of systems that are well described by this model are called *multi-indenture products*, and each level removed from the main system is called an *indenture level*. In military terms, the large systems (*e.g.*, an airplane) described in this paper are known as *Prime Equipment* (PE). Each PE is composed of one or more *Line Replaceable Units* (LRUs), and each LRU is composed of one or more *Shop Replaceable Units* (SRUs). Each of these terms represents an indenture level.

The most common approaches in the literature on developing a possible spare provisioning decision model are simulation and mathematical programming. Although these two problems (LORA and spare provisioning) are interdependent, they are seldom solved simultaneously due to the complicating nature of the relationships between spare levels and system availability. Thus in the spare parts literature and in practice, the approach to solve the joint LORA-Spares problem is to solve the LORA problem first (concerning the repair/replace decisions and the structure of the repair network), and subsequently solve the spare provisioning problem (concerning the number of spares held at each of the bases or depots).

Approach: The need to address LORA and the sparing problems simultaneously has attracted increased attention from the Department of National Defence. In this technical memorandum, a multi-objective approach is used to extend the problem from the traditional approach of finding the best sparing policy to achieve a predetermined operational availability to that of identifying the relationship between the operational availability and resource cost. In this regard, a multi-objective evolutionary algorithm (specifically the *Non-dominated Sorting Genetic Algorithm II*) is proposed that can identify non-dominated solutions to be used for a trade-off analysis. Multi-objective optimization is the process of simultaneously optimizing two or more conflicting objectives subject to certain constraints.

The following optimization objectives have been devised: minimization of total repair cost (*e.g.*, spare parts, spares transportation, spares storage) and maximization of operational availability. The solutions are made up of vectors which include the maximum number of LRUs and SRUs that would be stocked as well as their re-order points. In addition, the solution also includes the optimal repair network structure.

The approach uses a Monte Carlo simulation to generate scenarios based on a dataset which includes failures of the components and their associated times of failure. The average objective functions are computed at each genetic algorithm generation based on all generated scenarios.

Principal Results: Examples on a realistic dataset illustrate the type of trade-off analyses that can be carried out. The problem studied included thirty LRUs and thirty SRUs. The solutions from several different initial population sets were combined into a single aggregate data set and the non-dominated front of this data set was compared to the non-dominated front of each individual data set through the use of a hypervolume measure. All but two outlying fronts differed in this respect by no more than 8%. The two outlying fronts with large percent differences were due to their maximal downtime objective values compared to the corresponding value in the combined front. It was found that the large downtime objective value could be mainly attributed to one part in each of these two cases. Hence, the maintenance strategy is often more sensitive to parts with higher annual failure rates.

In the realistic example studied, it was determined that a small increase in stock (from one to two) of one of the LRUs would result in considerably lower system downtime (from 1.29 to 0.25 days) at a relatively small cost (0.2% difference).

Conclusions and Recommendations: Based on the results in this study, it is recommended that any part stocking strategy for SRUs and LRUs of DND PEs be examined through multiple lenses (and not purely on cost). The results in this study indicate that trade-offs can be found and provided to decision-makers. The ability to choose between several equally good (from a mathematical point of view) solutions allows the decision-maker to decide which objectives are more important in the context of available solutions.

Future work in this field could include:

- Testing the algorithm on larger, more complex problems to more closely approximate the decision space at DND;
- Testing the solutions in the non-dominated front on other scenarios generated from the same distribution in order to verify the robustness of the solutions; and
- Either incorporating vectorization and parallelization techniques in the code, or implementing the algorithm in a compiled programming language to decrease the computation required to generate results.

Sommaire

An Exposition on Solving the Joint Level of Repair Analysis-Spares Problem using a Multi-Objective Genetic Algorithm

Raman Pall, Slawomir Wesolkowski, Matthew Dozois ; DRDC CORA TM 2013–159 ; R & D pour la défense Canada – CARO ; septembre 2013.

Contexte : L'analyse du niveau de réparation (ANR) est souvent définie comme le processus qui consiste à déterminer si un composant défectueux doit être réparé ou jeté, et à quel endroit dans le réseau la réparation doit se faire le cas échéant. On doit aussi déterminer le nombre optimal de composants de rechange que l'on peut garder en inventaire pour une pièce d'équipement donnée.

La complexité de l'ANR tient au fait que la plupart des équipements sophistiqués ont une conception modulaire et qu'on en assure l'entretien par l'échange de pièces entre les unités de différents niveaux hiérarchiques. Au moment de la défaillance du système, on identifie le composant qui est à l'origine de la panne. Ce composant peut parfois être divisé en sous systèmes, ce qui permet d'imputer la panne à une pièce en particulier. Les types de systèmes qui sont décrits correctement par ce modèle sont désignés comme des *produits à niveaux d'intégration multiples*, et chaque niveau retiré du système principal est appelé *niveau d'intégration*. En termes militaires, on désigne les grands systèmes (p. ex. un aéronef) décrits dans ce rapport comme de l'*équipement principal* (EP). Chaque EP est composé d'un ou de plusieurs *éléments remplaçables en première ligne* (line replaceable units – LRU), et chaque LRU est composé d'un ou de plusieurs *éléments remplaçables en atelier* (shop replaceable units – SRU). Chacun de ces termes correspond à un niveau d'intégration.

La simulation et la programmation mathématique sont les méthodes qui reviennent le plus souvent dans la documentation portant sur l'élaboration d'un modèle décisionnel d'approvisionnement en pièces de rechange. Bien que ces deux problèmes (l'ANR et le stockage des pièces de rechange) soient interdépendants, ils sont rarement résolus simultanément en raison de la complexité des relations entre les niveaux de pièces de rechange et la disponibilité du système. C'est pourquoi dans les ouvrages sur les pièces de rechange et dans la pratique, on préconise de résoudre le double problème de l'ANR et du stockage des pièces de rechange en résolvant tout d'abord le problème de l'ANR (c'est à dire en ce qui regarde la décision de réparer ou de remplacer et la structure du réseau de réparation) et ensuite celui de l'approvisionnement en pièces de rechange (c'est à dire en ce qui concerne le niveau des stocks à chaque emplacement (bases ou dépôts)).

Méthode : Le ministère de la Défense nationale (MDN) porte une attention de plus en plus grande à la nécessité de s'attaquer simultanément au problème de l'ANR et à celui de l'approvisionnement en pièces de rechange. Dans ce document technique, nous proposons une approche multi objectifs pour modifier la façon de faire classique : au lieu de tenter de définir la meilleure politique de stockage qui garantirait un niveau de disponibilité opérationnelle préétabli, nous cherchons à définir le rapport entre la disponibilité opérationnelle et le coût des ressources. à cet égard, nous proposons un algorithme évolutionnaire multi objectifs (à savoir le *Non-dominated Sorting Genetic Algorithm*

II) capable d'offrir des solutions non dominées qui peuvent servir à l'analyse d'arbitrage. L'optimisation multi objectifs est un processus qui consiste à optimiser simultanément plusieurs objectifs contradictoires étant donné certaines contraintes.

Nous énonçons les objectifs d'optimisation suivants : réduire au maximum les coûts liés à la réparation (p. ex. pièces de rechange, transport des pièces, stockage des pièces) et maximaliser la disponibilité opérationnelle. Les solutions se composent de vecteurs qui comprennent le nombre maximum de LRU et de SRU qui seront stockés ainsi que leur seuil de réapprovisionnement. La solution comprend en outre la structure optimale du réseau de réparation.

Cet algorithme utilise une simulation de Monte Carlo pour créer des scénarios à partir d'un ensemble de données faisant état des pannes des composants et du moment de la panne. Les fonctions-objectifs moyennes sont calculées à chaque exécution de l'algorithme génétique sur la base de tous les scénarios générés.

Principaux résultats : Des exemples fondés sur un ensemble de données réaliste illustrent le genre d'analyses d'arbitrage qui peuvent être faites. Le problème examiné dans ce rapport comprend trente LRU et trente SRU. Nous avons regroupé les solutions tirées de plusieurs ensembles de populations de départ en un seul ensemble de données agrégées et avons comparé le front non dominé de cet ensemble de données au front non dominé de chacun des ensembles de données pris individuellement en nous servant d'une mesure d'hypervolume. Tous les fronts secondaires, sauf deux, présentent un écart de 8 % tout au plus. L'écart plus grand observé pour les deux fronts d'exception s'explique par la différence entre la valeur maximale de l'objectif de temps d'indisponibilité dans ces deux fronts et la valeur correspondante dans le front combiné. On a constaté que la valeur élevée de l'objectif de temps d'indisponibilité était principalement attribuable à une pièce d'équipement dans chacun des deux cas. En conséquence, la stratégie d'entretien est souvent plus sensible aux pièces qui ont un taux de défaillance annuel plus élevé.

Dans l'exemple réaliste étudié, nous avons déterminé qu'une légère augmentation (de une ou deux unités) du stock de l'un des éléments remplaçables en première ligne (LRU) aurait une forte incidence à la baisse sur le temps d'indisponibilité du système (le faisant passer de 1,29 jour à 0,25 jour), moyennant un coût relativement peu élevé (différence de 0,2 %).

Conclusions et recommandations : Suivant les résultats de cette étude, il est recommandé que toute stratégie de stockage des pièces concernant les SRU et les LRU de l'équipement principal (EP) du MDN soit examinée sous plusieurs angles (et non seulement sous l'angle des coûts). Les résultats de l'étude montrent qu'il est possible de trouver des solutions de compromis qui peuvent être mises à la disposition des décideurs. La capacité du décideur de choisir entre plusieurs solutions aussi bonnes les unes que les autres (d'un point de vue mathématique) lui permet de déterminer quels objectifs sont plus importants que d'autres étant donné les solutions qui s'offrent à lui.

Les futurs travaux de recherche dans le domaine pourraient comprendre les expériences suivantes :

– Tester l'algorithme sur des problèmes plus vastes et plus complexes afin d'évaluer avec plus de précision l'espace de décisions au MDN ;

– Tester les solutions du front non dominé sur d’autres scénarios tirés de la même distribution afin de vérifier la robustesse des solutions ;

– Intégrer des techniques de vectorisation et de parallélisation dans le code, ou exécuter l’algorithme dans un langage compilé afin de réduire le volume de calculs requis pour produire les résultats.

This page intentionally left blank.

Table of contents

List of tables

List of figures

Acknowledgements

The authors would like to acknowledge the following people who assisted our study:

- Mr. Derek Cranshaw (a co-operative student from the University of Waterloo employed by DRDC CORA in 2012), who assisted in the implementation of an earlier version of the model described herein; and
- Dr. Yaw Asiedu, DRDC, for numerous conversations regarding the LORA-Spares problem and for providing guidance on this work.

The author would like to acknowledge the following people who helped prepare this report for publication:

- Dr. A. Ghanmi (CJOC OS OR&A Team Lead), who reviewed the report; and
- Ms. C. Eisler (Force Readiness Analysis Team), who provided innumerable helpful comments in a peer review of the report.

1 Introduction

"There is one advantage to having nothing, it never needs repair."
— Frank Howard Clark (1888 – 1962), American screenwriter

1.1 Background and Context

An unfortunate fact of manufacturing is that products are often prone to failure. Whereas inexpensive products can be discarded and replaced upon failure, more expensive products are instead maintained by a repair by replacement policy: a failed component is removed from the product and replaced by a functioning spare part, if available. Otherwise, the replacement has to wait until a functioning component arrives.

Level of repair analysis (LORA) is an approach used during the design stage of complex equipment for the analysis of the cost effectiveness of competing maintenance strategies [1]. LORA is often defined as the problem of determining whether a component should be repaired or discarded upon its failure, and the location in the repair network to do such work. LORA is carried out as part of life cycle cost and cost of ownership analyses, and can play a significant role in minimizing these costs for capital equipment. This is especially important in defence, where maintenance requires complex support equipment and highly skilled personnel, and the unavailability of the equipment is especially undesirable [1].

The complexity of LORA lies in the fact that most of the complex equipment has modular design, and maintenance is carried out by exchanging parts at locations at different hierarchy levels (*i.e.*, there is a hierarchical relationship between the operating locations and their supporting depots). An issue of paramount importance in LORA is concerned with the locations of where failing items should be repaired or replaced in order to minimize the total support cost.

A related problem is the determination of the optimal number of spares for a given piece of equipment [2]. This is a major concern for many industrial organizations, including in defence and aircraft industries, due to the enormous capital spent on spares in these organizations every year. Spare provisioning plays a crucial role to ensure specified availability for a system – increasing the number of spares can sometimes increase availability at the expense of cost. As such, cost minimization and availability maximization are often competing objectives in this problem.

1.1.1 Multi-Indenture Products

In the context of LORA, upon failure of a system, the component which is determined to be responsible for the overall failure of the system is identified. Occasionally, this component can be broken down further into smaller subsystems, and the source of the failure can be traced to a single part. The types of systems that are well described by this model are called *multi-indenture products*, and each level removed from the main system is called an *indenture level*. All systems described in this paper are assumed to be multi-indenture products. The option to replace an item at any of its levels of indenture is a significant contributor to the complexity of the LORA problem.

In military terms, the large systems (*e.g.*, an airplane) described in this paper are known as *Prime Equipment* (PE). Each PE is composed of one or more *Line Replaceable Units* (LRUs), and each LRU is composed of one or more *Shop Replaceable Units* (SRUs). Each of these terms represents an indenture level. This concept is illustrated in Figure 1. In theory, the failure of a PE can be due to a failure at any indenture level, but in practice it is usually considered to be traceable to a single SRU.[1] Once the source of the failure is determined, the method of repair is to replace the failed component with a working one, if available. If there is no working component, the PE is out of commission until a replacement is made available.

As an example, consider a system in which a truck is a PE. One of the system's LRUs consists of a wheel containing two SRUs: the wheel's tire and rim. Suppose that the wheel fails. Either the entire wheel (LRU) can be replaced if available, or, if not, the problem is further investigated. Under the assumption listed above, the problem will be traceable to one (and only one) of the SRUs – either the tire, or its rim. The problematic SRU will then require replacement or repair. The choice of replacement at either indenture level adds to the complexity of the problem.

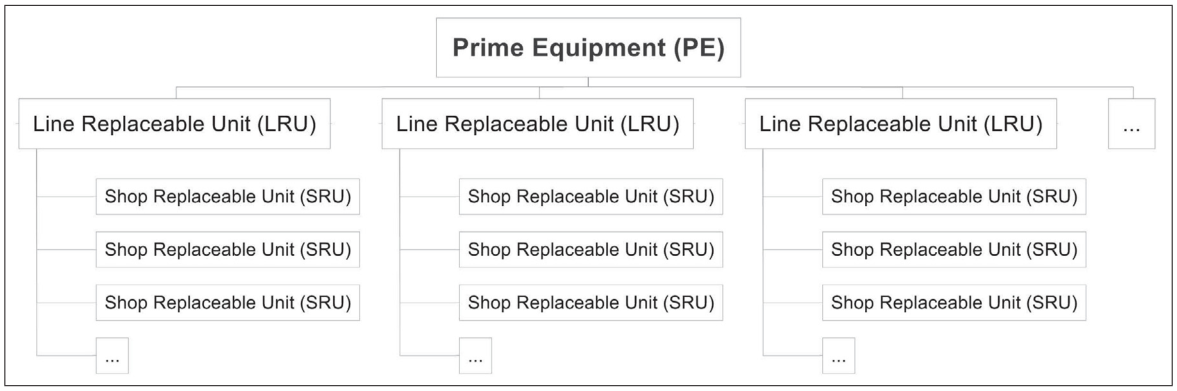

Prime Equipment (PE)
Line Replaceable Unit (LRU)
Line Replaceable Unit (LRU)
Line Replaceable Unit (LRU)
...
Shop Replaceable Unit (SRU)
Shop Replaceable Unit (SRU)
Shop Replaceable Unit (SRU)
...
Shop Replaceable Unit (SRU)
Shop Replaceable Unit (SRU)
Shop Replaceable Unit (SRU)
...
Shop Replaceable Unit (SRU)
Shop Replaceable Unit (SRU)
Shop Replaceable Unit (SRU)
...

Figure 1: *An illustration of the multi-indenture nature of the prime equipment.*

1.1.2 Structure of the Repair Network

All movement of broken components or restocking of working components is done through a defined system of supporting depots, called the *repair network*. Locating spares, and repair and test equipment close to each of the operating sites is cost-prohibitive. Therefore, central locations have certain resources which allow them to repair particular components which may not be able to be repaired at the operating sites, and are capable of resupplying locations where component stocks have been depleted. Once in operation, all locations have specific instructions on what to do if particular components are found to be faulty. Part of the goal of this paper is to determine the optimal repair network structure to minimize the overall support cost of the system. It is assumed that alternative repair solutions cannot be made on an *ad-hoc* basis.

The structure of the repair network forms a single tree with leaves representing locations called the *Forward Operating Bases* (FOBs), which are the only locations where PEs can fail. Each FOB has a

[1]Within this report, all failures in the PE can be associated with the failure of exactly one of its components, and not from the interaction of several failed components.

parent node from which it resupplies and to which it sends broken components, called *Intermediate Depots* (IDs). The number of IDs may range anywhere from one to the number of FOBs.[2] All IDs resupply from and send broken components to a single location called the *Central Depot* (CD), which is the root of the repair tree.[3] The CD is the farthest location from the FOB that a component can travel (in terms of links along the tree, not necessarily in terms of geographic distance). Once the component is received at the CD, it must be either repaired or replaced.

Each of these classes of nodes forms a single level in the repair network hierarchy, called an *echelon*. In principle, any number of echelon levels is possible. In practice however, it is usually limited to three. The set of operating sites, or FOBs, corresponds to the first such echelon, called echelon 1, the IDs form echelon 2, and the CD forms echelon 3. This concept is illustrated in Figure 2.

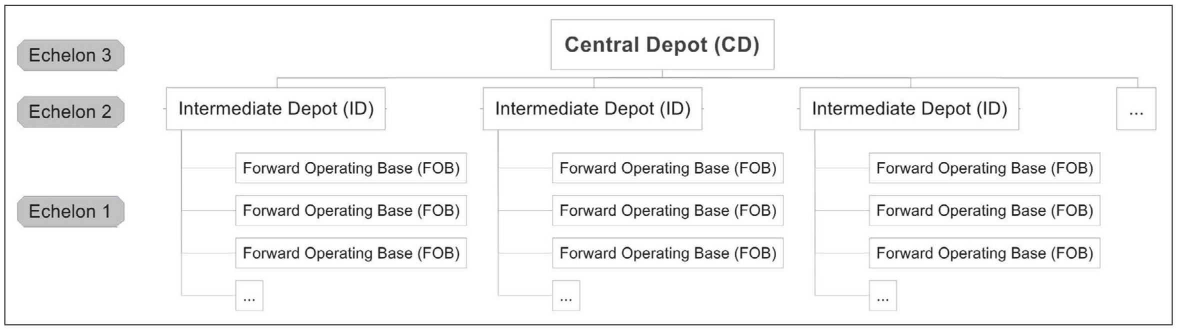


Figure 2: *An illustration of the repair network and the echelon concept.*

1.1.3 Description of the Problem

When a PE fails at an FOB, the decision of how to get the system back in working order has to be made. Regardless of what is to be done with the failed component, the first maintenance operation is to replace the failed LRU with a working one, if one is available at the base. If there is no replacement ready, the PE must remain out of commission at the FOB, incurring downtime cost, until a replacement is available.

Once the failed LRU is removed, it can be decomposed into its component SRUs at the base, or moved up to the Intermediate Depot associated with the FOB. This decision is based on the *Echelon Level of Repair* (ELOR) assigned to the LRU. The ELOR for a given part specifies the minimum echelon level at which the part can be repaired. As an example, a specific SRU may be repaired at an intermediate depot, and is thus given an ELOR of 2, while another may be more complex and can only be repaired at the central depot, and is thus given an ELOR of 3. All SRUs must have a ELOR greater or equal to that of their parent LRU, and all PEs are automatically assumed to have an ELOR of 1 (*i.e.*, PEs never leave FOBs). Moreover, each component is given a probability of being repaired at each of the possible echelon levels (subject to the constraint that this probability be 0 if the echelon level is lower than the ELOR of the component). The decision on the echelon level regarding where the failed component will be repaired is then handled on a stochastic basis. In

[2]There cannot be more IDs than the number of FOBs as each ID is associated to a non-zero set of FOBs.

[3]It is assumed in this paper that there is only one CD, and the repair network forms a single tree, as opposed to a set of trees (also known as a *forest*), wherein there would be T CDs, where T is the number of trees in the forest.

either case, the failed SRU(s) are then isolated and replaced with working ones as soon as they are available. The LRU is then put back together and kept at whichever location the repair took place.

Repaired LRUs are not immediately sent to the FOB where the failure was discovered. If an SRU is composed of parts, it can either be decomposed at that location or sent one echelon level higher in the network for repair, depending on the ELOR of the SRU. In either case, the failed parts are then isolated and replaced, and the repaired SRU stays at whichever location it was repaired.

Over time, the stock of various LRUs and SRUs will deplete at the FOBs, and accumulate at the intermediate and central depots. Thus, whenever the stock of a particular part at an FOB falls below some value, called the *reorder threshold*, the FOB makes a restock request with its associated intermediate depot for a number of parts that will bring the stock level of the FOB up to its *target stock level*. If the intermediate depot has the stock to satisfy the request, the parts are shipped immediately. Otherwise, the intermediate depot ships as many parts as it can, and a backorder[4] is generated at the FOB. The intermediate depots and the central depot have a similar relationship. All backorders are shipped as soon as parts are available at the supplying location, with the exception that locations will fulfill their own immediate part requirements before shipping backorders to its children locations.

1.1.4 Common Approaches to the Problem

In the spare parts literature and in practice, the approach to solve the joint LORA-Spares problem is to solve the LORA problem first (concerning the repair/replace decisions and the structure of the repair network), and subsequently solve the spare provisioning problem (concerning the number of spares held at each of the bases). Although the goal of this problem is to achieve a target availability of the PEs, a key feature of a number of models used in practice is that the focus is not on the maximization of the availability, but instead on the minimization of the expected number of backorders of parts at the bases. As a result of a backorder, a system is unavailable waiting for spares. Furthermore, when referring to optimality in the spare parts stocking problem and that efficient solutions are found, it is taken to mean that it is not possible to achieve a lower expected number of backorders without increasing the cost incurred.

The first LORA model was specified by Barros in 1998. Barros assumed that the same decisions are taken at all locations at one echelon level, that resources required to perform repairs are uncapacitated, and that each resource could repair all parts at a given level of indenture [1]. An integer linear programming model was used to approach the problem. Other approaches to the problem have included a branch-and-bound method [3] and a genetic algorithm approach [4]. More recently, Basten *et al.* proposed a mathematical programming formulation that generalizes these models by not requiring that resources be shared by parts, and allowing that different decisions may be taken at various locations at the same echelon level [5, 6].

The most common approaches in the literature on developing a possible spare provisioning decision model are simulation and mathematical programming. Of note, Sherbrooke [7] first applied mathematical programming to the spare parts inventory management problem in a multi-echelon

[4]A *backorder* occurs if a component is requested, but cannot be delivered to the requesting base immediately.

setting. In this work, he developed the METRIC model (Multi-Echelon Technique for Recoverable Item Control), a model which was used for a single-indenture problem using a greedy heuristic to optimize the base stock levels. Sherbrooke's work was extended by Muckstadt [8] in the MOD-METRIC model, which allowed for the more complex problem in which the system was comprised of two indenture levels. Further work by Graves [9] and Sherbrooke [10] resulted in more accurate approximations for the two-echelon problem under single-indenture and two indenture levels (respectively), known as the VARI-METRIC model. The heuristics employed to solve the sparing problem in these papers are often based on marginal analysis and require very tight restrictions on the resource-component relations in the associated LORA models. Exact evaluations of these models have appeared in the literature more recently, most notably by Rustenburg *et al.* [11], who consider the more general multi-echelon, multi-indenture problem, providing exact and approximate evaluations, as well as an overview of the related literature.

Although these two problems (LORA and spare provisioning) are interdependent, they are seldom solved simultaneously due to the complicating nature of the relationships between spare levels and system availability. The first paper in the literature that addressed the joint LORA and spares provisioning problem, by Alfredsson [12], considered two-echelon, single-indenture problems. Recently, Ilgin and Tunali [13] developed an approach for joint optimization of spare part provisioning and maintenance policies for an automotive factory by integrating simulation with a genetic algorithm. The simulation model of the manufacturing line was used as an input to the genetic algorithm as part of the fitness function. However, this paper was limited to optimization based solely on cost. Most recently, Basten *et al.*[14] have extended Alfredsson's optimization methodology with an exact algorithm that provably finds efficient solutions (meaning, once again, that it is not possible to achieve a lower expected number of backorders without increasing the cost incurred).

1.2 Study of the LORA-Spares Problem at DND

As mentioned previously, the combined problem of LORA and spare provisioning is of interest to the Department of National Defence (DND), as it can play a role in minimizing life cycle and maintenance costs for capital equipment, as well as increasing the availability of the equipment.

DND uses a tool known as the *OmegaPS Analyzer* [15] to approach the LORA-Spares problem. This tool uses marginal analysis techniques to approach the two interdependent problems using separate modules within the software. Experience has shown that results from OmegaPS Analyzer tend to overestimate system availability while underestimating spares requirements, which impacts both operational and financial planning at DND [16]. Hence, DND has a significant interest in developing models and methodologies to solve the combined LORA-Spares problem.

In 2010, Defence Research and Development Canada – Centre for Operational Research and Analysis (DRDC CORA) initiated funding to study the LORA-Spares problem through a departmental funding mechanism known as the Technology Investment Fund (TIF). More specifically, the goal of the TIF project was to develop new modelling and solution approaches that are applicable to generalized maintenance networks and more flexible spares allocation and resource utilization regimes. As an exploration into all possible approaches that could be used in the elucidation of this problem, a bevy of approaches are being considered: graph theory, network flow modelling, queuing theory, regression analysis, simulation modelling, decomposition techniques (*e.g.*, the Dantzig-Wolfe decomposition) and evolutionary optimization techniques [16].

The model described herein is an output of this TIF project. Other studies associated to the TIF are numerous. They include an overview by Sakr and Asiedu of the most popular models currently available in the literature to solve the LORA and Spares problems (individually or simultaneously) [17]. Ghaddar, Sakr and Asiedu studied the combined problem when formulated as a single mixed integer non-linear optimization model to explicitly capture the interdependency between the repair network and sparing inventory decisions. The resulting optimization model is solved through decomposition optimization approaches, a simulation model, and a genetic programming-based symbolic regression methodology [18, 19]. In yet another study, Zhang and Asiedu constructed a variant of Sherbrooke's METRIC model to deal with small fleet sizes, based on the use of truncated distributions for calculating the expected backorder of parts [20].

1.3 Scope of the Analysis and the Structure of the Report

In this technical memorandum, we detail a multi-objective approach to extend the problem from the traditional approach of finding the best sparing policy to achieve a predetermined operational availability to that of identifying the relationship between the operational availability and resource cost. In this regard, an evolutionary algorithm is proposed that can identify non-dominated solutions to be used for a trade-off analysis.

Optimization objectives are twofold: the minimization of repair costs (*e.g.*, spare parts, spares transportation, spares storage) and the maximization of availability (or equivalently, the minimization of downtime of prime equipment). The approach developed uses a Monte Carlo simulation to generate different scenarios based on a dataset which includes the expected failures of the equipment and their associated probabilities. The average values of the objective functions are computed at each iteration of the genetic algorithm based on all generated scenarios (once generated, the set of scenarios remains static).

This paper is organized as follows. The multi-objective optimization method as it pertains to the LORA-Spares problem is explained in Section 2, with additional background materials in Annexes A to C. Section 3 describes applications of the technique and the results that were obtained. The paper terminates in Section 4 with recommendations and concluding statements.

2 The Approach

A multi-objective approach was used to extend the problem from the traditional approach of finding the best sparing policy to achieve a predetermined operational availability to that of identifying the relationship between the operational availability and resource cost. In this regard, an evolutionary algorithm is proposed that can identify non-dominated solutions to be used for a trade-off analysis. A brief introduction to the concept of multi-objective optimization is provided in Section 2.1, with a more comprehensive explanation provided in Annex A.

A Monte Carlo simulation model was created, implemented in the *Matlab* mathematical programming environment, to generate scenarios from a dataset detailing the expected failures of the equipment and their associated probabilities [21]. The simulation was run multiple times to determine the average values of the objectives for each population member. A multi-objective genetic algorithm was then used to perform the optimization analysis. More specifically, the *Non-dominated Sorting Genetic Algorithm II* (NSGA-II), constructed by Deb *et al.* [22, 23], was used to perform the optimization. A brief description of NSGA-II's mechanics as it relates to the current problem is provided in Section 2.2.

2.1 Multi-Objective Optimization

Multi-objective optimization is the process of simultaneously optimizing two or more conflicting objectives subject to certain constraints. In general, when the objectives in a problem are conflicting, they prevent simultaneous optimization of each objective. In other words, generally there is no single solution that simultaneously optimizes each objective. As such, one seeks to find a set of solutions that cannot be improved with respect to any objective without worsening at least one other objective.

The concept of *dominance* is used to determine this set of solutions: one solution is said to dominate another if the first is at least as good at optimizing each objective as the second, and is better than the second on at least one of the objectives. If neither solution dominates the other, we say that the solutions are *non-dominated.*

A solution is said to be *Pareto-optimal* if it is not dominated by any other solution in the solution space. These solutions are precisely those which cannot be improved with respect to any objective without worsening at least one other objective. The set of all feasible non-dominated solutions is referred to as the *Pareto-optimal set*, and for a given Pareto-optimal set, the corresponding objective function values in the objective space are called the *Pareto front* [24, 25]. A more complete discussion on multi-objective optimization can be found in Annex A.

2.2 The Genetic Algorithm

Genetic algorithms (GAs) are algorithms which mimic the biological process of evolution to solve search and optimization problems [24, 25, 26, 27, 28]. Each solution to the optimization problem is given a fitness score which relates to its ability to optimize the problem in question. Those solutions with the highest fitness are used to generate more solutions in a subsequent iteration of the algorithm (known as a *generation*). As the algorithm progresses, its non-dominated front gets closer and closer to the Pareto-optimal front for the problem.[5] A detailed primer on genetic algorithms (GAs) is provided in Annex B for the interested reader.

By applying a genetic algorithm to our combined problem of level of repair analysis and spares provisioning, we can obtain a set of solutions making up the non-dominated front in objective space (*i.e.*, the space with axes labelled “Resource Cost”, and “Operational Availability”). In this way, we obtain a range of non-dominated solutions varying in cost and operational availability instead of simply obtaining the lowest cost solution at a specified minimum operational availability – which is the current output of common techniques in the literature [13].

2.2.1 NSGA-II

The specific GA used in this work is the *Non-dominated Sorting Genetic Algorithm II* (NSGA-II), constructed by Deb *et al.* in 2002 [22, 23]. The NSGA-II algorithm was chosen as a result of its relatively low computational complexity, retention of the fittest solutions from one iteration of the algorithm to the next (a property known as *elitism* in GA parlance), and preservation of diversity amongst the solutions, *i.e.*, ensuring that they do not all converge to the same set of solutions, but instead to a large range of solutions.

The NSGA-II algorithm is among the most popular of all GAs for these reasons. In particular, it has seen usage in Canadian defence applications in Air Force fleet mix optimization problems [29, 30]. A brief description of NSGA-II’s mechanics as it relates to the current problem is provided in Annex C. In brief, each solution to the problem is comprised of the information representing the stock levels, reorder thresholds, and ELOR of the components, as well as information regarding the structure of the repair network.

2.3 The Model

The particulars of the model are described in this section – the objectives, encoding of the chromosome, and the main operations within the genetic algorithm.

2.3.1 Objectives

The objectives that we consider in our model are the following:

- Minimization of the total monetary cost of the maintenance strategy; and
- Maximization of the total availability of the system.

The costs included in the model that are captured by the first objective are defined as follows:

[5] It is worthwhile noting that it is difficult to ascertain whether a GA has converged to the Pareto-optimal set of solutions after the algorithm has run for several generations [24, 27]. For these reasons, one often speaks of the final non-dominated set of solutions found via the algorithm (after a given number of generations) in place of the Pareto-optimal set.

- *Shipping Cost:* The cost to ship failed parts either to a location higher in the hierarchy (*i.e.*, sending a failed part to a parent for repair), or to a location lower in the hierarchy (*i.e.*, a resupply of a repaired or spare part). This cost is incurred every time a part is shipped.
- *Holding Cost:* The daily costs of holding parts overnight at FOBs and IDs, which is directly proportional to the number of components at each base.
- *Repair Cost:* The cost of repairing a particular part at a particular echelon.

The algorithm sums all of these costs for each scenario and averages over all scenarios. Naturally, those solutions with the lowest average overall cost are the fittest with respect to this objective.

The second objective of the GA is to maximize the total availability of the system or, equivalently, to minimize the downtime of all PEs is the system. Each PE type has a "downtime cost" associated with it, which determines how much the system will suffer should the PE be out of commission. The total downtime cost is given by the sum over all PEs of the number of days each PE is out of commission, multiplied by the downtime cost of that PE. The algorithm averages the downtime costs over all scenarios, and the solutions with lowest average overall cost are the most fit with respect to this objective.

2.3.2 Chromosome Representation

In this paper, no PE has more than four indenture levels and the repair network has exactly three echelon levels. The solution chromosomes consist of two segments: the first detailing information about the components, and the second detailing information about the locations. For the first segment, each component has a target stock level and reorder threshold for each of the three echelon levels. In addition, each component has an ELOR. Thus, this segment of the chromosome has $7n$ genes, where n is the number of components in the system. The second segment details the intermediate depots from which each of the FOBs obtains their resupplies (*i.e.*, the parent location of each of the FOBs). The length of this segment is m, where m is the number of FOBs in the system. Hence the total length of the chromosome is $7n+m$.

2.3.3 Scenario Generation

The fitness functions evaluate the efficiency of a repair management strategy based on a number of randomly generated failure scenarios. Once these scenarios are generated, they are set and used repeatedly in the objective function calculation (*i.e.*, the scenarios are not re-generated with each GA generation). Concretely, the same scenarios are used in each of the iterations of the GA , i.e., the looping through the GA encompasses the Monte Carlo scenarios. Performing the loops in this order ensures that each member of the population is evaluated against the same set of scenarios for fair comparison of the objective function averages.

The algorithm used to generate a scenario operates as follows: for each component, a random number of failures is sampled from a Poisson distribution with mean equal to the *Average Yearly Failure* (AYF) of that component. This results in a total of $\sum_{i=1}^{n} f_i$ failures in the year.

Each individual failure is randomly assigned a specific date in the year for the event (an integer from 1 to 365), as well as a location (*i.e.*, FOB) where the failure occurred. The probability of failing on a certain date is uniform across all dates, while the probability of failing at the specific FOBs is taken as an input to the model. The resulting list of failures, sorted by ascending date, is a single scenario. The model is run using many randomly generated scenarios (*i.e.*, using a Monte Carlo approach) to determine the optimal maintenance strategy (with respect to the structure of the repair network and the number of spares held at each of the locations).

2.3.4 Population Initialization

A population of size p is randomly generated at the start of the GA. The genes of each chromosome are generated in the following way:

- The target stock level for each component at each echelon is chosen randomly from 1 to some theoretical upper bound. The bound for each component is chosen by using the cumulative probability of that component failing up to a given number of times. Setting the bound in this way allows the elimination of solutions with an excess number of spare components which would cause an increase in overall cost without reducing the repair time of the PE.
- The reorder threshold for each component at each echelon is chosen from 0 to one less than the target stock level.
- The ELOR of each component is chosen. Since a component must have an ELOR no lower than that of its parent, the algorithm decides on the ELOR of all ancestors of a component before choosing an ELOR for that component. The ELOR of a component is randomly chosen between the ELOR of its parents and 3. If the component is a PE, it is automatically given an ELOR of 1.
- Each FOB is randomly assigned to an intermediate depot from which it resupplies and to which it sends its items for repairs.

Note that in the analysis of a real system, the steps above may not necessarily applicable – for example, the components would most likely already be associated to pre-determined ELORs.

2.3.5 Crossover and Mutation

The binary tournament selection procedure is used to choose parents for the crossover operator [23]. Four solutions are chosen at random from the population, and the fitter of the first two and the fitter of last two are selected as the parent solutions for the crossover operation. This is repeated p times to create $2p$ parents in the mating pool.

The standard crossover (random exchange of elements in the two parent chromosomes) is used for mating parts of the chromosome related to the stock levels and reorder points as well as the location of the FOBs.

Given the tree structure for the ELOR, the crossover operation must preserve the property that all components have a higher ELOR than their parent components. To do so a pivotal node is randomly selected. The tree with the higher ELOR at the pivotal node is labeled parent A, while the other is labeled parent B. The child tree inherits the entire branch starting from the pivotal node, including the pivotal node, from parent A, and all other nodes from parent B. In additional detail, the crossover operation is divided into the following three stages:

- *Stock level stage:* The algorithm carries out the following crossover procedure for each component.
  - Assuming there are N parts, the program randomly chooses an integer n, where $1 < n < N$.
  - The algorithm chooses n random parts from parent 1, and the remaining $N - n$ parts from parent 2.
  - The stock levels and reorder points of each part are then passed to the child from each corresponding parent.
- *ELOR stage:* The multi-indenture structure of a system can be thought of as a tree, with the PE as the root node and the parts as the leaves. The crossover operation must preserve the property that all components have a higher ELOR than their parent components. To do this, the following crossover procedure is carried out for each system tree.
  - A random node in the tree, called the pivotal node, is selected.
  - The tree with the higher ELOR at the pivotal node is labeled parent A, while the other is labeled parent B.
  - The child tree inherits the entire branch starting from the pivotal node, including the pivotal node, from parent A, and all other nodes from parent B.
- *FOB stage:* No assumptions are made regarding which forward operating bases are associated with which intermediate depots. Therefore, for each operating base, the child simply chooses randomly whether to inherit the intermediate depot from parent 1 or parent 2.

Like the crossover function, mutation has to be done in sections, since different sections of the chromosome mutate in different ways. Each section mutates in the following way.

- *Stock level stage:* The standard mutation operator [23] is used to randomly mutate the stock level for each component at each echelon. The stock level is mutated to a random number within its allowable range (from 1 to some theoretical upper bound) if the mutation is successful. If the stock level for a part successfully mutates, the reorder point mutates to ensure that the reorder point remains below the stock level for all components at all echelons.

- *ELOR stage:* The IDs associated to each of the FOBs are randomly mutated. Given that each component must always have a higher ELOR than its parent component, components may only mutate their ELORs to values above or equal to the ELOR of their parent node, and below or equal to the minimum of the ELOR of their children. The order in which parts are considered for mutation is randomized to ensure that the structure of the tree after mutation is not biased. All PEs maintain an ELOR of 1.

- *FOB stage:* No assumptions are made regarding which FOBs are associated with which Intermediate Depots. Therefore, for each successful mutation, the FOB is randomly assigned an Intermediate Depot.

2.3.6 Termination of the Algorithm

The algorithm terminates after a number of generations specified by the analyst. The algorithm is run a specified number of times, called the number of *trials* or *runs*. After the last run is complete, the solutions in the final generations of each run are combined together and the non-dominated front of the combined solution set is found. This set, while not necessarily the Pareto-optimal front, is the output of the model.

3 Applications of the Technique

In this section, the search space is defined in Section 3.1, and results are included for the model for two examples – the first is a small prototypical example, for which the true optimal values could be determined via exhaustive search; and the second represents a large, realistic example which would correspond to the size (and associated repair network) of a system in use by DND. These examples are provided in Sections 3.2 and 3.3, respectively.

3.1 Defining the Search Space

As mentioned in Section 2.3.4, a maximum upper bound needs to be set at each echelon of the repair network for each component. The bound for each component was set using the cumulative probability of the component failing a given number of times. The cumulative probability distribution for an SRU is given by

$$P(0 \leq x \leq n) = P(0) + \cdots + P(x) + \cdots + P(n)$$

where x is a positive integer representing the number of failures of the part in the scenario, and $P(x)$ is the probability of such an occurrence. $P(x)$ is determined by the Poisson process with a rate parameter equal to the part's AYF. The cumulative probability distribution of an LRU failing is also a Poisson process. Since the failures of the SRUs are assumed to be independent of one another, the Poisson probability distributions can be multiplied together to show the rate parameter of the Poisson process of the LRU is the summation of each of the AYF for each SRU that composes the LRU [31]. The value of n was then chosen such that the cumulative failure probability would be equal to or greater than 99.9%. Since less than a 0.1% chance of a part failing more than n times in a year exists, stocking more than n parts would unnecessarily increase the holding cost without providing a benefit to the time objective except in extreme cases.

3.2 A Validated Protoypical Example

In order to illustrate the use of the methodology, we provide the results generated for a prototypical example consisting of a small repair network comprised of one central depot, two intermediate depots, and four FOBs. The algorithm was implemented in *Matlab* and run for 500 generations with a population size of 100, under 10 failure scenarios. The small size of this example enabled the true optimal values to be easily determined via exhaustive search, which illustrate the validity of the algorithm.

The initial population was randomly generated, and the mutation rate used was 0.15. There were twelve types of parts in the example; of which two are PEs, four are LRUs, and six are SRUs. The hierarchy of the parts is depicted in Figure 3, where their level of indenture is shown (several parts have sub-components common to other parts).

The costs of shipping parts from one location to another ranged from \$0 to \$10,000 depending on the part number and the locations involved. Further, the shipping costs and times between location pairs were chosen such that the optimal maintenance strategy was obvious with little analysis. Each SRU was assumed to fail with equal probability at each FOB. Only the SRUs were given non-zero

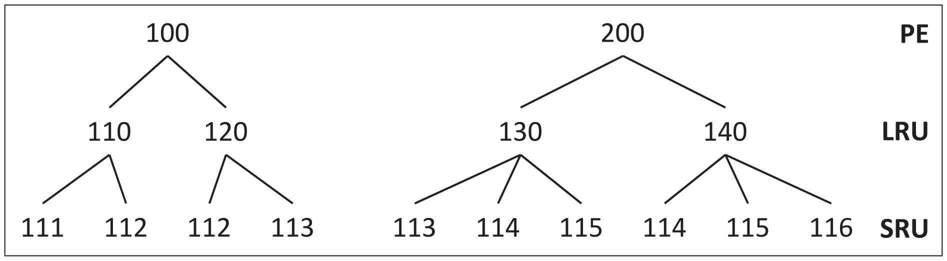


Figure 3: *Hierarchy of the parts in the example.*

probabilities of failure (specified by their average yearly failure values) – meaning that the LRUs and PEs did not fail directly in the example, and all failures could be traced to the failure of a single SRU. Finally, the failure of each part was assumed to contribute equally to the downtime of the PE regardless of the cause of failure of the PE.

It was found that the algorithm converged to a non-dominated front of four solutions, as can be seen in Figure 4 and Table 1. Moreover, it was found that this set of solutions was precisely the the Pareto front for the example in question, illustrating that the algorithm converged to the required front as desired.

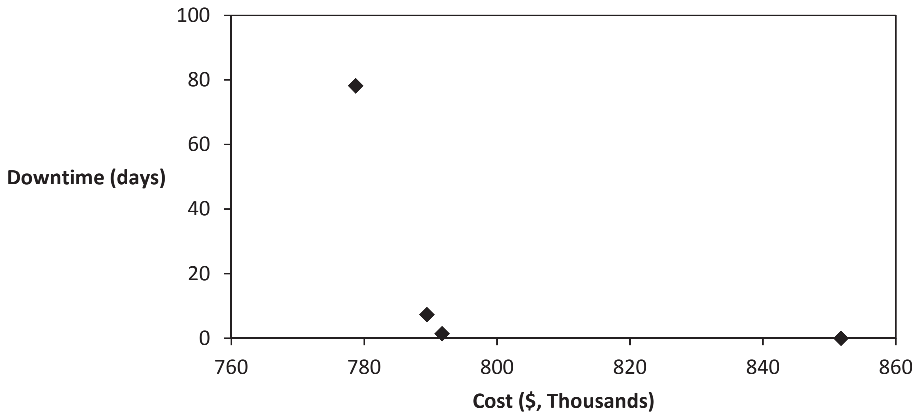


Figure 4: *The non-dominated front found in the example, consisting of four solutions.*

Table 1: *Details on the non-dominated front found in the example, consisting of four solutions.*

Cost ($)	Downtime (days)
778,700	78.2
789,419	7.3
791,697	1.4
851,751	0

Note in particular that a methodology solving the problem at the minimum cost would have found only the first solution (with a cost of $778,700, and an associated system downtime of 78.2 days);

whereas another single-objective mathematical programming methodology requiring a constraint on the downtime (say, downtime < 10 days), such as Ilgin and Tunali's methodology [13], would find only the second solution (with a cost of $789,419, and an associated system downtime of 7.3 days). The advantage of the methodology presented here is that it illustrates to the decision-maker many of the feasible solutions to the problem along the non-dominated front, granting them additional flexibility in their decision making process. For example, they would find that an increase of 0.3% in cost from the second solution (moving from $789,419 to $791,697) would yield a solution decreasing the downtime incurred by 80.8% (going from 7.3 days to 1.4 days of downtime incurred).

3.3 A Large, Realistic Example

With only 10 parts, an exhaustive search provided a reasonable method for finding the optimum solution in the prototypical example. However, for realistic examples which would interest DND, an exhaustive search is not feasible due to the size of the search space.

In this example, the repair network of the system consisted of one central depot, two intermediate depots and four FOBs. The part network had one PE, thirty LRUs and thirty SRUs as shown in Figure 5. Note that the part with ID 603 is the PE, and parts with ID 1 to 60 are the LRUs and SRUs. Those parts with no children (such as 31, 32, and 5) are SRUs, and the intermediate nodes (such as 1, 2, and 3) are LRUs.

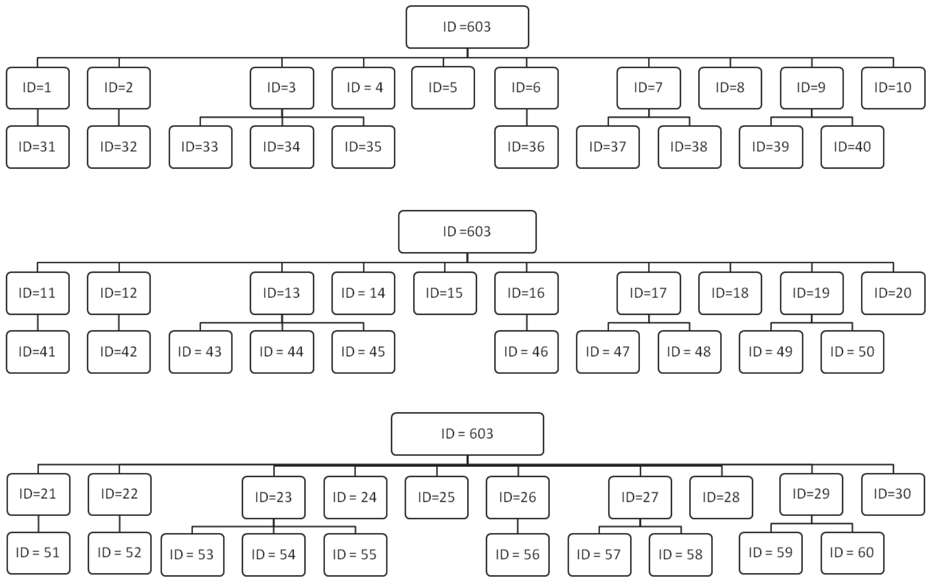


Figure 5: *Part hierarchy of the realistic example.*

3.3.1 The Search Space

Given the possible repair and maintenance structure, the size of the search space was examined. The number of reorder combinations that exist for each stock threshold of each part is equal to the stock threshold for that part. Furthermore, the number of stock threshold combinations for each part at each echelon is equal to the theoretical upper bound set using the Poisson distribution. Consequently, the number of possible solutions for each part at each echelon is equal to the theoretical upper bound multiplied by the summation of 1 to the theoretical upper bound. The number of combinations between all parts at each echelon was then calculated by multiplying the resultant values together. Afterwards, the numbers of combinations at each echelon were multiplied together since the values are independent. The total number of combinations given maximum stocking values and reorder points of the parts was approximated to be 10^{311} based off of the AYF of the parts.

Each LRU can also be repaired at any of the three echelons, and the SRUs used in the LRU can only be repaired at the echelons equal to or greater to the parent LRU which means six possible combinations exist for the ELOR for any given SRU and parent LRU. The approximated value was then raised to the power of thirty since the number of LRUs and SRUs are each equal to thirty. The operating bases can be supplied from any of the eight facilities with each of the operating bases being able to differ in the supply location. Hence, the number of combinations from possible supply locations was approximated to be the number of combinations possible from selecting four facilities as shipping locations from the eight total facilities. Once all the aforementioned values were calculated, the values were multiplied together to obtain the approximated search space size of 10^{338} which is too large to be searched exhaustively in a reasonable timeframe.

3.3.2 Inputs for the GA

To search the space, the genetic algorithm was run for 20 different initial populations, each iterated through 500 generations with an initial population size of 100, and for 100 failure scenarios. The mutation rate of the genetic algorithm was set to 0.15.

The cost of shipping parts from one location to another varied from \$38 to \$102 and was dependent upon both the part and the shipping locations. The time of shipping a part from one location to another varied from 8 days to over 21 days and was again dependent on both the part and the shipping locations. SRUs and LRUs which were not composed of SRUs were given non-zero probabilities of failures. Damaged parts could also be reshipped to a higher echelon than the repair echelon if the failure was too severe to be repaired at the part's designated repair echelon. The failure of each part contributed equally to the downtime of the PE.

3.3.3 Combining the Fronts

The solutions from the different initial population sets were combined into a single aggregate data set and the non-dominated front of this data set was compared to the non-dominated front of each individual data set through a hypervolume measure [32]. A reference point for the hypervolume was chosen to be twice the maxima of the time and fitness objective values (32 days, 3.35 million dollars) of the non-dominated fronts examined.

The relative differences between the hypervolume measures of each individual run and the the aggregate front are listed in Table 2. The extrema of the final generation in each run specified in the table, represented by their maximum and minimum values along the objectives (measured in \$ for the Cost objective, and days for the Downtime objective). The relative differences in the hypervolume between the aggregate Non-Dominated Front (NDF) and the individual NDFs in each run are also provided in the table, and expressed as percentages.

The median value was found to be 2.63%; the minimum, 0.27%; and all but two outlying fronts differing by no more than 8%. The two outlying fronts with large percent differences – 57.9% and 71.5%, respectively, and highlighted in Table 2 – were due to their maximal downtime objective values (12.3 and 16.0 days, respectively), compared to the maximal downtime objective value of 1.34 days in the combined front. The large downtime objective value in each of these two cases could be mainly attributed to one part. For the solution with the time objective value of 12.3 days, the LRU with ID 26 contributed to 11.4 out of the 12.3 days, and for the solution with an objective value of 16.0 days the LRU with ID 12 contributed to 15.5 out of 16.0 days. Both of the SRUs which compose these LRUs had annual yearly failure rates of 1.98 failures per year, the highest failure rate out of any of the SRUs. Hence, the maintenance strategy can be sensitive to parts with high annual failure rates. A decision-maker's role becomes of practical importance when outlying solutions exist because the solutions can be easily disregarded during the decision process if undesirable.

3.3.4 The Non-Dominated Solutions

The combined front consisted of 60 unique chromosomes which mapped to 10 unique objective values. Each objective function reached a maxima when the other objective reached a minima and vice versa. The supporting operating base, intermediate depot, and central depot repair network was the same for both the optimal cost and optimal time solutions.

Both extreme solutions also had fairly evenly distributed echelon levels of repair. The minimal-time solution had 19 parts repaired at the FOBs, 16 at the IDs, and 25 at the CD; whereas the minimal cost solution had 21 repaired at the operating bases, 19 at the intermediate depots, and 20 at the central depot. These results show that some parts being repaired at the central depot may be providing a benefit to the time objective but increasing the cost objective. Moreover, the optimal cost solution also reduced overall cost by stocking more parts across the echelons than in the optimal time solution (specifically, a total maximum of 309 compared to 301 parts).

Some of the stocked parts affected the objective values more than others. Parts such as the LRU with ID 38 differed in every stocking level between solutions whereas the SRU with ID 11 had the same stocking levels for each solution. The differences in the optimal objective functions along the non-dominated front emphasize how the repair network and maintenance strategy can be adjusted based on the goals of the decision maker.

The non-dominated front in the combined data set is shown in Figure 6, and the corresponding objective values are shown in Table 3. The decision making process could dictate the use of the global maxima and minima if one of time or cost takes much higher precedence. However, in actual situations, a trade-off between cost and downtime may be more beneficial. For example, a decision maker could choose to reduce the downtime objective by 80% (from 1.29 to 0.25 days) while only increasing the cost objective by 0.2% (from 1.440 to 1.442 million dollars). The large reduction in downtime was found to be due to the maintenance strategy for one particular LRU, which contributed 1.09 days to the solution. By doubling the stock of this LRU at the first echelon from one to two, the cost was slightly increased while achieving a large reduction in the PE downtime.

Table 2: *Details on the NDF found in the 20 runs for the example – including the extrema of the NDF in each run, and their relative hypervolume differences with the aggregate NDF. The two outlying fronts are also identified.*

	Extrema of the final generation				Hypervolume
Run	min(Cost)	max(Downtime)	max(Cost)	min(Downtime)	Difference (%)
15	1,540,700	0.29	1,627,000	0.00	0.27
20	1,555,000	0.54	1,628,400	0.00	0.35
1	1,583,500	0.11	1,627,100	0.00	0.62
10	1,520,000	0.47	1,614,700	0.00	1.15
5	1,585,100	0.27	1,699,000	0.00	1.77
16	1,471,800	1.13	1,603,800	0.00	1.82
12	1,510,100	0.45	1,734,900	0.00	1.82
11	1,547,100	0.02	1,598,300	0.00	2.63
9	1,535,500	0.39	1,627,200	0.00	2.01
14	1,433,700	1.34	1,540,200	0.00	2.27
4	1,518,900	0.30	1,547,400	0.00	3.77
13	1,516,900	0.08	1,636,000	0.00	4.27
3	1,489,900	0.06	1,766,000	0.00	4.99
6	1,477,900	0.37	1,532,700	0.00	5.45
17	1,500,200	0.06	1,597,400	0.00	5.47
2	1,506,700	1.76	1,589,300	0.00	6.22
7	1,467,100	0.09	1,541,400	0.00	6.50
8	1,514,700	2.20	1,569,700	0.00	7.38
19	1,503,100	12.33	1,622,600	0.00	57.86*
18	1,530,800	15.98	1,627,400	0.00	71.45*
Aggregate	1,433,732	1.34	1,523,721	0.00	*: *outlying front*

This reduction shows that the maintenance strategy can be adjusted to accommodate parts which contribute significantly to the objectives.

Thus, these non-dominated solutions can be used for a trade-off analysis in which the decision maker would be able to decide between solutions depending on political, budgetary, and other motivations. This result indicates that a multi-objective GA may be a viable method of addressing the LORA-Spares problem.

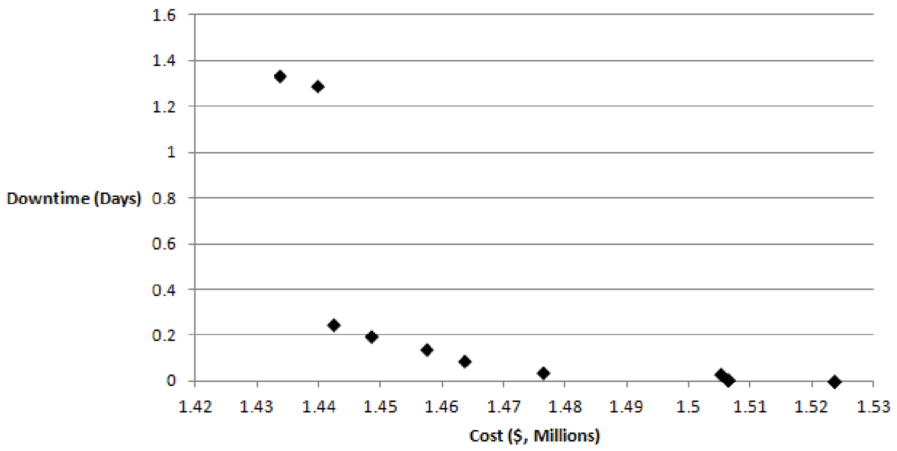


Figure 6: *The non-dominated solutions in the example found after combining the solutions found in the various runs of the GA.*

Table 3: *Details on the non-dominated front found in the large example, consisting of ten solutions.*

Cost ($)	Downtime (days)
1,433,732	1.34
1,439,842	1.29
1,442,452	0.25
1,448,562	0.20
1,457,400	0.14
1,463,509	0.09
1,476,284	0.04
1,505,119	0.03
1,506,475	0.01
1,523,721	0.00

4 Conclusions and Recommendations

The need to address LORA and the sparing problem simultaneously is a problem of significant interest to DND. In this technical memorandum, we extended the state of the art in the study of the joint LORA-Spares problem by proposing to approach this problem through an application of a multi-objective GA, where both cost and availability are objectives to be optimized. Two examples were provided in which non-dominated solutions were identified that can be used for a trade-off analysis. In the realistic example, it was determined that a small increase in stock (from one to two) of one of the LRUs would result in considerably lower system downtime (from 1.29 to 0.25 days) at a relatively small cost (0.2% difference).

From a technical point of view, a key aspect of generating solutions in a reasonable amount of time was the application of ceilings on the maximum required stock for each of the parts in order to significantly limit the search space (which is still quite large).

4.1 Recommendations

Based on the results in this study, it is recommended that any part stocking strategy for SRUs and LRUs of DND PEs be examined through multiple lenses (and not purely on cost). The results in this study indicate that trade-offs can be found and provided to decision-makers. The ability to choose between several equally good (from a mathematical point of view) solutions allows the decision-maker to decide which objectives are more important in the context of available solutions.

4.2 Future Work

Future work in this field could include:

- Testing the algorithm on larger, more complex problems to more closely approximate the decision space at DND;
- Testing the solutions in the non-dominated front on other scenarios generated from the same distribution in order to verify the robustness of the solutions to similar new scenarios; and
- Either incorporating vectorization and parallelization techniques in the *Matlab* code, or implementing the algorithm in a compiled programming language (*e.g.*, *C* or *C++*) to decrease the computation time.

References

[1] Barros, L. (1998), The optimization of repair decisions using life-cycle cost parameters, *IMA J. Math. App. in Business & Industry*, 9, 403–413.

[2] Basten, R. (2009), Designing logistics support systems: Level of repair analysis and spare parts inventories, Ph.D. thesis, University of Twente, Enschede, the Netherlands.

[3] Barros, L. and Riley, M. (2001), A combinatorial approach to level of repair analysis, *European Journal of Operational Research*, 129(2), 242–251.

[4] Saranga, H. and Kumar, U. D. (2006), Optimization of aircraft maintenance/support infrastructure using genetic algorithms – level of repair analysis, *Annals of Operations Research*, 143(1), 91–106.

[5] Basten, R., Schutten, J., and van der Heijden, M. (2009), An efficient model formulation for level of repair analysis, *Annals of Operations Research*, 172(1), 119–142.

[6] Basten, R., Schutten, J., and van der Heijden, M. (2011), A minimum cost flow model for level of repair analysis, *International Journal of Production Economics*, 133(1), 233–242.

[7] Sherbrooke, C. (1968), METRIC: a multi-echelon technique for recoverable item control, *Oper. Res.*, 16, 122–141.

[8] Muckstadt, J. (1973), A model for a multi-item, multi-echelon, multi-indenture inventory system, *Management Science*, 20(4), 472–481.

[9] Graves, S. (1985), A multi-echelon inventory model for a repairable item with one-for-one replenishment, *Management Science*, 31(10), 1247–1256.

[10] Sherbrooke, C. (1986), VARI-METRIC: improved approximations for multi-indenture, multi-echelon availability models, *Oper. Res.*, 34, 311–319.

[11] Rustenburg, W., van Houtum, G., and Zijm, W. (2003), Exact and approximate analysis of multi-echelon, multi-indenture spare parts systems with commonality, In Shanthikumar, J., Yao, D., and Zijm, W., (Eds.), *Stochastic Modelling and Optimization of Manufacturing Systems and Supply Chains*, pp. 143–176, Boston (MA): Kluwer.

[12] Alfredsson, P. (1997), Optimization of multi-echelon repairable item inventory systems with simultaneous location of repair facilities, *European Journal of Operational Research*, 99, 584–595.

[13] Ilgin, M. and Tunali, S. (2007), Joint optimization of spare parts inventory and maintenance policies using genetic algorithms, *Int. J. Adv. Manuf. Technol.*, 34, 594–604.

[14] Basten, R., van der Heijden, M., and Schutten, J. (2012), Joint optimization of level of repair analysis and spare parts stocks, *European Journal of Operational Research*, 222, 474–483.

[15] Pennant Canada Limited. (2008). OmegaPSAnalyzer: Users reference guide release 4.0.

[16] Asiedu, Y., Technology Investment Fund (TIF) Project Proposal: Combining Level Of Repair Analysis And Spares Provisioning, dated 28 June 2010.

[17] Sakr, N. and Asiedu, Y. (2012), An Overview of Recent Literature on Level-of-Repair Analysis and Spare Parts Stocking, (Technical Memorandum DRDC CORA CP 2012-027) DRDC – Centre for Operational Research and Analysis.

[18] Ghaddar, B. and Asiedu, Y. (2012), An Integrated Level of Repair Analysis and Spare Parts Stocking Model, (Technical Memorandum DRDC CORA TM 2012-248) DRDC – Centre for Operational Research and Analysis.

[19] Ghaddar, B., Sakr, N., and Asiedu, Y. (2012), Spare Parts Stocking Analysis using Genetic Programming, *European Journal of Operational Research*. Preprint submitted on May 29, 2012.

[20] Zhang, R. and Asiedu, Y. (2012), An Approach to Sparing Analysis for a Finite Working Item Population, In for Modeling & Simulation International, T. S., (Ed.), *Proceedings of the 2012 Autumn Simulation Multi-Conference – AutumnSim'12, October 28-31, 2012*, San Diego, CA, USA.

[21] Wojtaszek, D. and Wesolkowski, S. (2011), Multi-objective Evolutionary Optimization of a Military Air Transportation Fleet Mix with the Flexibility Objective, In *2011 IEEE Computational Intelligence Symposium on Security and Defense Applications, Paris, France*.

[22] Srinivas, N. and Deb, K. (1994), Multiobjective optimization using nondominated sorting in genetic algorithms, *J. Evol. Comp.*, 2, 221–248.

[23] Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002), A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II, *IEEE Trans. Evol. Comp.*, 6, 182–197.

[24] Deb, K. (2001), Multi-Objective Optimization using Evolutionary Algorithms, Chichester, England: John Wiley & Sons.

[25] Konak, A., Coit, D. W., and Smith, A. E. (2006), Multi-objective optimization using genetic algorithms: A tutorial, *Reliability Engineering and System Safety*, 91, 992–1007.

[26] Haupt, R. L. and Haupt, S. E. (1998), Practical Genetic Algorithms, New York: John Wiley & Sons.

[27] Whitley, D. (1993), A Genetic Algorithm Tutorial, (Technical Report CS-93-103) Colorado State University (Department of Computer Science), Fort Collins, CO 8052.

[28] Fonseca, C. M. and Fleming, P. J. (1993), Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion, and Generalization, In Forrest, S., (Ed.), *Genetic Algorithms: Proceedings of the Fifth International Conference*, San Mateo, CA: Morgan Kaufmann.

[29] Mazurek, M. and Wesolkowski, S. (2009), Fleet mix computation using evolutionary multiobjective optimization, In *IEEE Symposium on Computational intelligence in multi-criteria decision-making*, pp. 46–50.

[30] Stuive, L., Wesolkowski, S., and Ghanmi, A. (2010), Tactical Fleet Mix Computation Using Multiobjective Evolutionary Optimization, In *2010 IEEE Congress on Evolutionary Computation*, pp. 1–7.

[31] Adelson, R. (1966), Compound Poisson Distributions, *Operational Research Society*, 17(1), 73–75.

[32] While, L., Hingston, P., Barone, L., and Husband, S. (2006), A Faster Algorithm for Calculating Hypervolume, *IEEE Transactions on Evolutionary Computation*, pp. 29–38.

[33] Sawaragi, Y., Nakayama, H., and Tanino, T. (1985), Theory of Multiobjective Optimization, Vol. 176 of *Mathematics in Science and Engineering*, Orlando, FL: Academic Press Inc.

[34] Holland, J. (1985), Adaptation in natural and artificial systems, Ann Arbor: University of Michigan Press.

[35] Dawkins, R. (1976), The Selfish Gene, New York: Oxford University Press.

[36] Rudolph, G. (1994), Convergence Analysis of Canonical Genetic Algorithms, *IEEE Transactions on Neural Networks*, 5(1), 96–101.

[37] Schaffer, J. (1985), Multiple objective optimization with vector evaluated genetic algorithms, In *Proceedings of the international conference on genetic algorithm and their applications*.

[38] Fonseca, C. and Fleming, P. (1993), Multiobjective genetic algorithms, In IEE, (Ed.), *IEEE colloquium on "Genetic Algorithms for Control Systems Engineering" (Digest No. 1993/130), 28 May 1993*, London, UK.

[39] Murata, T. and Ishibuchi, H. (1995), MOGA: multi-objective genetic algorithms, In IEEE, (Ed.), *Proceedings of the 1995 IEEE international conference on evolutionary computation, 29 November–1 December*, Perth, WA, Australia.

[40] Zitzler, E. and Thiele, L. (1999), Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach, *IEEE Trans. Evol. Comput.*, 3(4), 257–271.

[41] Zitzler, E., Laumanns, M., and Thiele, L. (2001), SPEA2: improving the strength Pareto evolutionary algorithm, Technical Report Swiss Federal Institute Techonology, Zurich, Switzerland.

[42] Knowles, J. and Corne, D. (1999), The Pareto archived evolution strategy: a new baseline algorithm for Pareto multiobjective optimisation, In IEEE, (Ed.), *Proceedings of the 1999 congress on evolutionary computation – CEC99, 6–9 July 1999*, Washington, DC, USA.

[43] Corne, D., Knowles, J., and Oates, M. (2000), The Pareto envelope-based selection algorithm for multiobjective optimization, In Springer, (Ed.), *Proceedings of sixth international conference on parallel problem solving from Nature, 18–20 September, 2000*, Paris, France.

[44] Zitzler, E., Deb, K., and Thiele, L. (2000), Comparison of multiobjective evolutionary algorithms: empirical results, *Evol. Comput.*, 8(3), 173–195.

[45] Coello, C. (1999), A comprehensive survey of evolutionary-based multiobjective optimization techniques, *Knowl. Inform. Syst.*, 1(3), 269–308.

[46] Coello, C. (2000), An updated survey of GA-based multiobjective optimization techniques, *ACM Comput. Surv.*, 32(2), 109–143.

[47] Nojma, J., Narukawa, K., Kaige, S., and Ishibuchi, H. (2005), Effects of Removing Overlapping Solutions on the Performance of the NSGA-II Algorithm, In Springer, (Ed.), *Evolutionary Multi-criterion Optimization Lecture Notes in Computer Science*, pp. 341–354.

Annex A: Multi-Objective Optimization

In mathematics, the concept of *optimization* refers to choosing the best element from some set of available alternatives. In the simplest case, this means solving problems in which one seeks to minimize or maximize a function by altering the values of a set of variables from within an allowed range. Many real world problems, however, have usually two or more conflicting objectives which cannot be adequately analyzed using single-objective optimization algorithms. These problems are referred to as *multi-objective* problems.

Multi-objective optimization is the process of simultaneously optimizing two or more conflicting objectives subject to certain constraints. Multi-objective optimization problems can be found in a variety of fields. These include product and process design; finance; economics; vehicle design; or more generally, wherever optimal decisions need to be taken in the presence of trade-offs between two or more conflicting objectives [24, 25, 33].

In mathematical terms, a multi-objective problem can be written as follows:

$$\min_x [f_1(x), f_2(x), \ldots, f_n(x)] \text{ s.t. } \begin{array}{l} g_j(x) \le 0,\ 1 \le j \le m \\ x_l \le x \le x_u \end{array} \tag{A.1}$$

The functions $\{f_i \mid 1 \le i \le n\}$ are called the *objectives*; the functions $\{g_j \mid 1 \le j \le m\}$ are called the *constraints*; and x_l and x_u are lower and upper bounds on the value of x, respectively. Note that all objectives in equation (A.1) are assumed to be of the minimization type (as opposed to the maximization type). A maximization type objective can be converted to a minimization by multiplying the objective by -1.

In general, the objectives in such a problem are conflicting, preventing simultaneous optimization of each objective. In other words, generally there are no single solutions that simultaneously optimize each objective. As such, one generally seeks to find a set of solutions that cannot be improved with respect to any objective without worsening at least one other objective. The concept of *dominance* is used to determine this set of solutions.

Given feasible solutions x_1 and x_2 for a multi-objective problem[6], x_1 is said to *dominate* x_2, written $x_1 \succ x_2$, if and only if x_1 is at least as good at minimizing each objective as x_2, *i.e.*, $f_i(x_1) \le f_i(x_2)$ for all i, and there is at least one objective at which it does so better (there exists an i^* such that $f_{i^*}(x_1) < f_{i^*}(x_2)$). If x_1 does not dominate x_2 and if x_2 does not dominate x_1, we say that the solutions are *incomparable* or *non-dominated*.

A solution is said to be *Pareto-optimal* if it is not dominated by any other solution in the solution space.[7] These solutions are precisely those which cannot be improved with respect to any objective without worsening at least one other objective. The set of all feasible non-dominated solutions is referred to as the *Pareto-optimal set*, and for a given Pareto-optimal set, the corresponding objective function values in the objective space are called the *Pareto front* [24, 25].

[6] A solution is said to be *feasible* if it does not violate any of the constraints of the problem.

[7] The term *Pareto* in *Pareto-optimal* is named after Vilfredo Pareto (1848–1923), an Italian economist who used the concept in his studies of economic efficiency and income distribution.

This page intentionally left blank.

Annex B: Genetic Algorithms

There are two classical approaches to multiple-objective optimization. One is to combine the individual objective functions into a single function. Another involves moving all but one objective to the constraint set.

In the first method, the analyst is required to provide a series of weights that represent the decision-maker's preferences. This may prove to be significantly difficult, as it requires all objectives be measured on the same scale. Moreover, small changes in the weights can often lead to quite different solutions. In the second method, a bound must be established for each of the former objectives which have been moved to the constraint set, which can be rather arbitrary. In both cases, these methods return a single solution rather than a set of non-dominated solutions. Hence, classical approaches to multiple-objective optimization suffer from a major failing: a lack of a global perspective.

More recently, a new class of search and optimization algorithms which do not suffer from this failing have been garnering significant attention. These methods, which mimic the biological process of evolution to solve search and optimization problems, were first developed by John Holland in 1975 and are known as *evolutionary algorithms* techniques [34].

The most popular of these techniques are known as *genetic algorithms* (GAs) [24, 25, 26, 27, 28], which are algorithms inspired by Darwin's *survival of the fittest* principle [35]. Put simply, this principle states that the life expectancy of an individual is dependent on the fitness of the individual. This causes genetically strong individuals the opportunity to produce more offspring than their genetically weaker brethren. Moreover, an individual's offspring tends to inherit some of their traits.

Random changes may occur in genes during reproduction that may cause speciation events. If these changes prove beneficial to the offspring, they may be retained in future generations. Conversely, detrimental changes are eliminated by natural selection. In the long run, the combination of genes in a species causing it to be genetically strong become dominant in their population.

In terminology of GAs, a solution is called an individual or a *chromosome*. Chromosomes are made of discrete units called *genes*. Each gene controls one or more features of the chromosome. The map between the solution space and the chromosomes is called an *encoding*. The set of chromosomes under examination is referred to as a *population*.

In most GAs, the population is initialized with a random set of chromosomes. A function known as the *fitness function* assigns to each solution a fitness score, which is directly related to the objective function(s) of the search and optimization problem.

Two operators are used to generate new solutions from existing ones: *crossover* and *mutation*. The crossover operator combines two chromosomes (referred to as the *parents*) to form new chromosomes (referred to as the *offspring*, or *children*). The parents are selected among existing chromosomes in the population, with a preference towards choosing those with higher fitness. This ensures that offspring have a stronger chance of inheriting the genes which encourage fitness. The mutation operator introduces random changes into chromosomes by altering the values of some of their genes. Very few chromosomes are affected when the mutation operator is applied to the population, and very few of the genes of the affected individuals are altered.

These two operators play critical and distinct roles in GAs. The crossover operator encourages population convergence towards fit chromosomes, and the mutation operator introduces genetic diversity into the population, avoiding the pitfall of convergence to local optima.

Finally, the chromosomes are selected for survival into the next stage of the population (referred to as a *generation*). In most GAs, the fitness of an individual determines the probability of its survival for the next generation.

It is worthwhile noting that it is difficult to ascertain whether a GA has converged to the Pareto-optimal set of solutions after the algorithm has run for several generations [24, 27]. For these reasons, one often speaks of the final non-dominated front found via the algorithm after a given number of generations in place of the Pareto-optimal front. In order to assess the convergence of the non-dominated front to the Pareto-optimal front, the concept of *hypervolume* is often used [32].

Hypervolume is a measure of the size of the portion of the objective space that is dominated by the non-dominated front of solutions. Such a calculation produces a single scalar figure which allows for simple comparisons between the non-dominated fronts produced by genetic algorithms. The hypervolume of each front is calculated with respect to a reference point. In Figure B.1. is shown a visual representation for the hypervolume of a front which consists of two objectives.

One may also note that, for the vast majority of problems, convergence of GAs is not a significant issue, especially in algorithms in which the fittest solutions are retained between subsequent generations [36].

B.1 A List of Notable Multi-Objective GAs

There are numerous well-known multi-objective GAs used in academia and in industry. Generally, these algorithms differ in their fitness assignment procedures; retention of the fittest solutions from one iteration of the algorithm to the next (a property known as *elitism* in GA parlance); and their approaches to ensure diversification in the solutions.

The first multi-objective GA, called vector evaluated GA (or VEGA), was proposed by Schaffer in 1985 [37]. Since that time, several other multi-objective GAs have seen substantial use, including the following:

- the Multi-objective Genetic Algorithm (MOGA) [38];
- the Random Weighted Genetic Algorithm (RWGA) [39];
- the Nondominated Sorting Genetic Algorithm (NSGA) [22];
- the Strength Pareto Evolutionary Algorithm (SPEA) [40];
- the improved SPEA (SPEA2) [41];

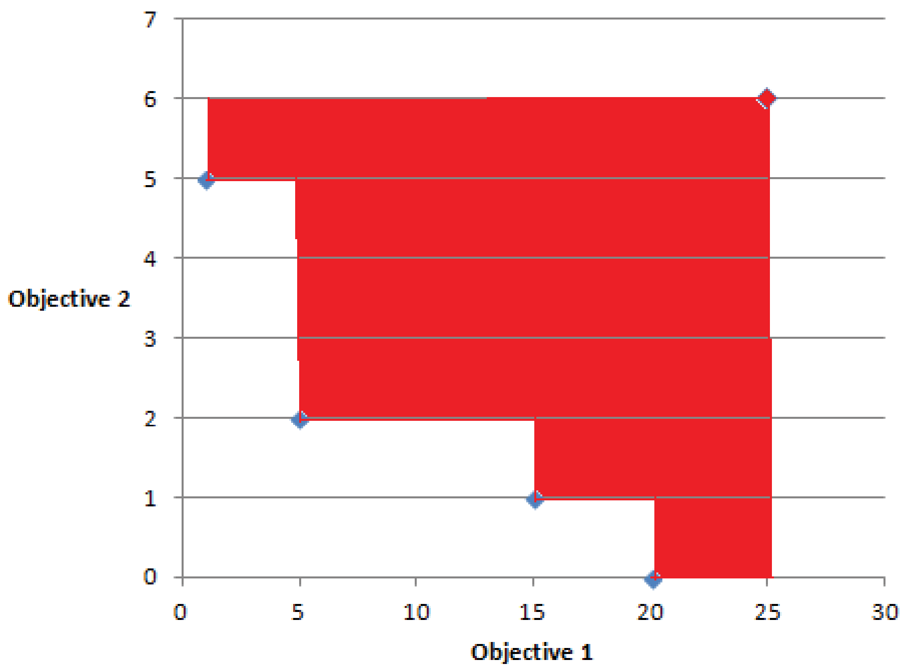


Figure B.1: *The hypervolume of non-dominated front for a two objective optimization. In this example, the hypervolume is computed as the area between the reference point (marked in red) and the non-dominated front.*

- the Pareto-Archived Evolution Strategy (PAES) [42];
- the Pareto Envelope-based Selection Algorithm (PESA) [43]; and
- the Fast Nondominated Sorting Genetic Algorithm (NSGA-II) [23].

Several survey papers [44, 45, 46] have been published that provide additional detail on these multi-objective GAs.

This page intentionally left blank.

Annex C: The Steps in NSGA-II

A brief description of NSGA-II's mechanics as it relates to the current problem is provided in this annex.

C.1 The Initial Population

The first generation consists of a randomly generated set of solutions (not necessarily feasible), which are used to seed the search space. This initial population has a very large impact on the resulting solution. Some initial populations favor certain areas of the Pareto Front. As a result, the genetic algorithm is typically run several times with different initial populations in the hopes of uncovering the entire front.

C.2 The Tournament and Mating Pool

To construct a mating pool, solutions are randomly selected from the population in groups of two, and the fitter solution of the two is added to the mating pool. This process (commonly known as a *tournament*) is repeated until the mating pool is as large as the population size. Note that a solution may be added multiple times to the mating pool. The fitness function used to evaluate solutions emphasizes both high values of the objective functions, as well as diversity of the solutions in the population.

C.3 Values of the Objective Functions and Front Numbers

It is obvious that a solution x_1 that dominates another solution x_2 would be preferred over x_2. However, in the more general case where the solutions are incomparable it is not obvious how one would compare the two.

A solution to this problem is to group our solutions into non-dominated fronts. This means partitioning the solutions into groupings such that every pair of solutions in the same front are incomparable, and every solution in one grouping will dominate, or be dominated by, all solutions in another front. The fronts are then numbered such that any member in front j will be dominated by at least one member in front $j-1$.

To create these fronts, all non-dominated solutions from the population are placed in front 1. Those which are non-dominated among the remaining solutions are placed in front 2. The process is iterated until the entire population has been placed into subsequent fronts.

To illustrate this procedure, consider the following example with two objectives, in which the goal is to minimize both objectives. Suppose we have 5 solutions in our population having the following objective values: $(2,1)$, $(2,6)$, $(3,2)$, $(4,5)$, and $(1,2)$. The solutions which dominate all other solutions and which are not dominated by any others form the first front. Hence $(1,2)$ and $(2,1)$ form Front 1. Of the remaining solutions, $(3,2)$ and $(2,6)$ are non-dominated, and are thus form Front 2. The only remaining solution is $(4,5)$, and so it forms Front 3. This process is presented graphically in Figure C.1.

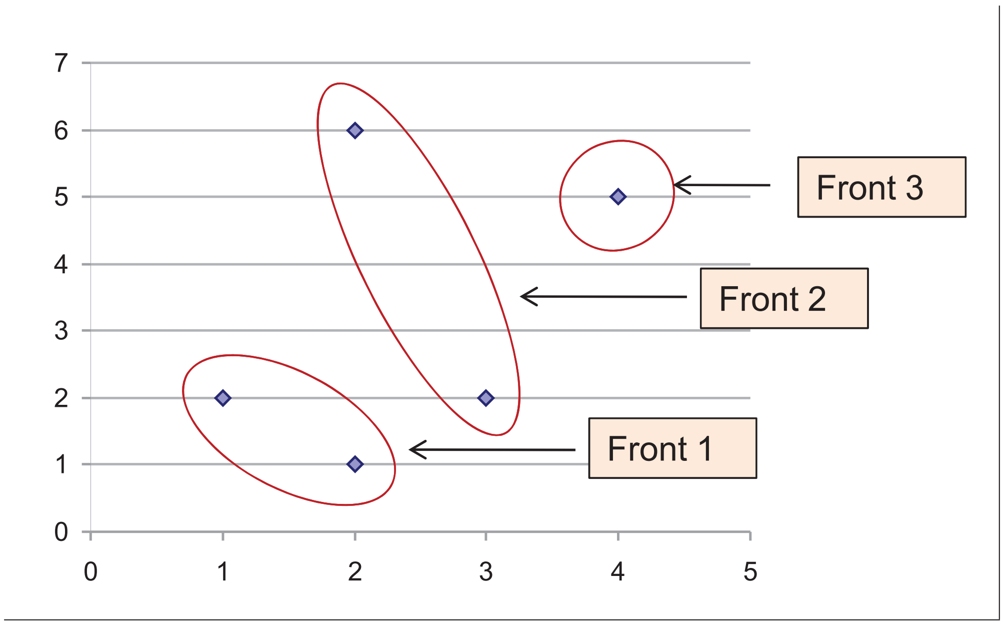


Figure C.1: *Partitioning the population in the example into non-dominated fronts.*

C.4 Diversity, Elimination of Redundant Solutions, and Crowding Distance

To prevent our population from clustering onto one solution on the Pareto front, redundant solutions are removed and the idea of *crowding distance* are used. The removal of solutions which have the same chromosomes ensures the uniqueness of the chromosome of each individual in the population, thereby improving the diversity and convergence of the Pareto front [47].[8]

The goal of using the idea of crowding distance is to have many solutions on the Pareto front with large distance to adjacent solutions as opposed to multiple solutions occupying the same portion of the Pareto front. Choosing the solutions with the largest distance from one another along the front ensures that the decision-makers are given as diverse a set of optimal solutions as possible.

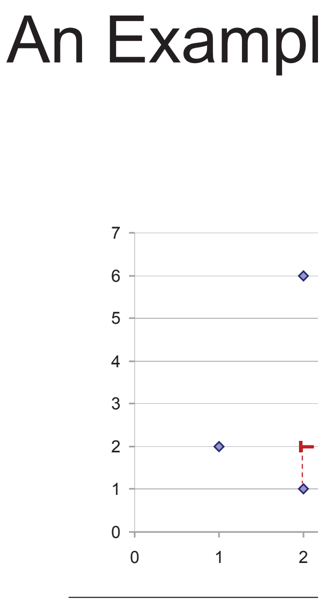
To calculate this metric, all solutions in the population are sorted according to each objective f_i. The crowding distance for each solution is defined as the sum of all the distances between the solution's closest neighboring points across all objective functions. The extreme points on each objective are given a crowding distance of infinity. Written mathematically, it is calculated as

$$CD(x) = \sum_{i=1}^{n} \frac{f_i(x_i^+) - f_i(x_i^-)}{\max_{y \in P} f_i(y) - \min_{y \in P} f_i(y)}$$

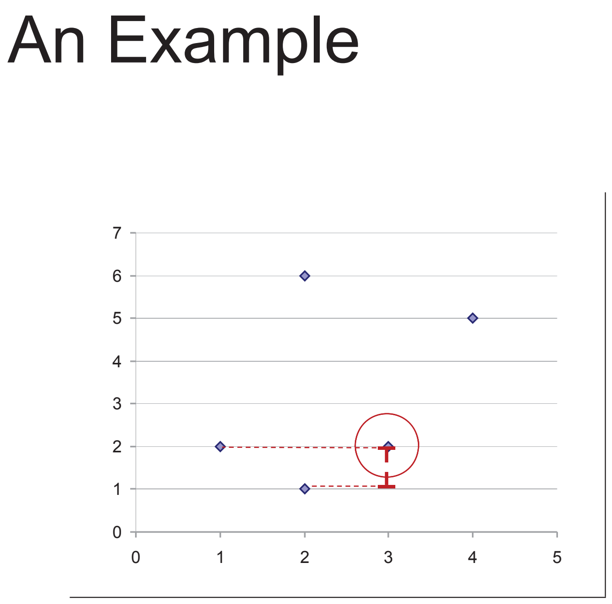
where $\{f_i \mid 1 \leq i \leq n\}$ are the objective functions, P is the set of solutions (*i.e.*, the population), and x_i^+ and x_i^- are the closest neighbours of x according to objective f_i.

[8]Note that this action does not contradict the fact that a solution may be added to the mating pool multiple times, as the removal of solutions to increase diversity (using the crowding distance) is performed when the fittest solutions are selected, which occurs *after* the mating and mutations step (which modifies the mating pool).

Continuing the example from the previous section, suppose we wish to calculate the crowding distance of point $(3,2)$. Along the first objective, its neighbors are $(2,1)$ and $(4,5)$. The minimum value for the first objective is 1 and the maximum is 4. Hence, the point's crowding distance is



(a) Calculation of the poir
the direction of the first ol
nearest neighbours in this (



(b) Calculation of the point $(3,2)$'s crowding distance in the direction of the second objective. Note that the point's nearest neighbours in this objective are $(2,1)$ and $(1,2)$.

Figure C.2: *An example of the calculation of the crowding distance for point* $(3,2)$. *The calculation along the first objective is shown in (a), while its calculation along the second objective is shown in (b).*

C.5 Fitness Hierarchy

In order to to compare the relative fitness of solutions, the front numbers of the solutions are examined. If one has a superior (*i.e.*, lower) front number, that solution is chosen as the fitter solution. If they have the same front number, the crowding distance of the solutions is used to determine the relative fitness of the solutions. The solution with the higher crowding distance is chosen as the fitter solution. In the unlikely event that they share the same crowding distance as well, the solutions are deemed to be equally fit.

C.6 Mating and Mutation

Following the tournament using the fitness relation and the creation of the mating pool, the solutions undergo the mating and mutation processes, referred to as *reproduction*. Solutions in the mating pool are paired up (randomly) and either reproduce or mutate.[9]

[9]The choice of the fraction of solutions which are formed through mating or mutation is rather arbitrary, and is chosen by the analyst. Generally, in the literature it has been found that the choice of mutation rate that minimizes convergence time (in terms of the number of generations required) is highly problem-specific.

These two processes provide the algorithm with ways to explore new areas of the search space. Mating produces children that are combinations of their parents, while mutations produce children that are slightly different than *one* of their parents. The general idea is that mating will try to make stronger members of the population by combining traits from strong parents. Mutations will help us find new solutions by modifying existing traits in the hopes of bringing new and stronger traits to the table.

C.7 Selection of the Next Generation

The final step in the genetic algorithm process is selecting the fittest solutions.

After mating and mutation, the population size doubles. To remedy this, the top half of the population with respect to their fitness level is retained, and the lesser half is killed. This method preserves only the strongest individuals and ensures that good solutions are never replaced with worse ones.

The entire process is iterated until the algorithm converges on a group of solutions, according to pre-defined convergence criteria. These criteria can include having the mean or standard deviation of the population's objective values not change between successive generations, no improvement in the size of the non-dominated front, computing for a set number of generations, or several other reasons [26].

List of Acronyms & Abbreviations

ANR	analyse du niveau de réparation
AYF	Average Yearly Failure
CANOSCOM	Canadian Operational Support Command
CD	Central Depot
CDS	Chief of the Defence Staff
CF	Canadian Forces
CFAWC	Canadian Forces Air Warfare Centre
CJOC	Canadian Joint Operations Command
CORA	Centre for Operational Research and Analysis
DND	Department of National Defence
DRDC	Defence Research and Development Canada
ELOR	Echelon Level of Repair
EP	équipement principal
FC	Forces canadiennes
FOB	Forward Operating Base
GA	Genetic Algorithm
ID	Intermediate Depot
LORA	Level of Repair Analysis
LRU	Line Replaceable Unit
MDN	ministère de la Défense nationale
METRIC	Multi-Echelon Technique for Recoverable Item Control
MND	Minister of National Defence
MOGA	Multi-objective Genetic Algorithm
NDHQ	National Defence Headquarters
NDF	Non-Dominated Front
NSGA	Non-dominated Sorting Genetic Algorithm
NSGA-II	Non-dominated Sorting Genetic Algorithm II
OS	Operational Support
PAES	Pareto-Archived Evolution Strategy
PE	Prime Equipment

PESA	Pareto Envelope-based Selection Algorithm
RWGA	Random Weighted Genetic Algorithm
SPEA	Strength Pareto Evolutionary Algorithm
SRU	Shop Replaceable Unit
TIF	Technology Investment Fund
VEGA	Vector Evaluated Genetic Algorithm

DOCUMENT CONTROL DATA

(Security classification of title, body of abstract and indexing annotation must be entered when document is classified)

1. ORIGINATOR (The name and address of the organization preparing the document. Organizations for whom the document was prepared, e.g. Centre sponsoring a contractor's report, or tasking agency, are entered in section 8.) Defence R&D Canada – CORA Dept. of National Defence, MGen G. R. Pearkes Bldg., 101 Colonel By Drive, Ottawa ON K1A 0K2, Canada	2a. SECURITY CLASSIFICATION (Overall security classification of the document including special warning terms if applicable.) UNCLASSIFIED 2b. CONTROLLED GOODS (NON-CONTROLLED GOODS) DMC A REVIEW: GCEC APRIL 2011

3. TITLE (The complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S, C or U) in parentheses after the title.)

An Exposition on Solving the Joint Level of Repair Analysis-Spares Problem using a Multi-Objective Genetic Algorithm

4. AUTHORS (Last name, followed by initials – ranks, titles, etc. not to be used.)

Pall, R.; Wesolkowski, S.; Dozois, M.

5. DATE OF PUBLICATION (Month and year of publication of document.) September 2013	6a. NO. OF PAGES (Total containing information. Include Annexes, Appendices, etc.) 54	6b. NO. OF REFS (Total cited in document.) 47

7. DESCRIPTIVE NOTES (The category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.)

Technical Memorandum

8. SPONSORING ACTIVITY (The name of the department project office or laboratory sponsoring the research and development – include address.)

Defence R&D Canada – CORA
Dept. of National Defence, MGen G. R. Pearkes Bldg., 101 Colonel By Drive, Ottawa ON K1A 0K2, Canada

9a. PROJECT OR GRANT NO. (If appropriate, the applicable research and development project or grant number under which the document was written. Please specify whether project or grant.) N/A	9b. CONTRACT NO. (If appropriate, the applicable number under which the document was written.)
10a. ORIGINATOR'S DOCUMENT NUMBER (The official document number by which the document is identified by the originating activity. This number must be unique to this document.) DRDC CORA TM 2013–159	10b. OTHER DOCUMENT NO(s). (Any other numbers which may be assigned this document either by the originator or by the sponsor.)

11. DOCUMENT AVAILABILITY (Any limitations on further dissemination of the document, other than those imposed by security classification.)

(X) Unlimited distribution
( ) Defence departments and defence contractors; further distribution only as approved
( ) Defence departments and Canadian defence contractors; further distribution only as approved
( ) Government departments and agencies; further distribution only as approved
( ) Defence departments; further distribution only as approved
( ) Other (please specify):

12. DOCUMENT ANNOUNCEMENT (Any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). However, where further distribution (beyond the audience specified in (11)) is possible, a wider announcement audience may be selected.)

Unlimited

13. ABSTRACT (A brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual.)

Level of repair analysis (LORA) is often defined as the problem of determining whether a component should be repaired or discarded upon its failure, and the location in the repair network to do such work. A related problem is the determination of the optimal number of spare components for a given piece of equipment. The most common approaches in the literature on developing a possible spare provisioning decision model are simulation and mathematical programming. Although these two problems (LORA and spare provisioning) are interdependent, they are seldom solved simultaneously due to the complicating nature of the relationships between spare levels and system availability.

The need to address LORA and the sparing problems simultaneously has attracted increased attention from the Department of National Defence (DND). In this technical memorandum, the use of a multi-objective genetic algorithm (specifically the Non-dominated Sorting Genetic Algorithm II) is proposed to solve this problem, with optimization objectives as minimizing repair costs (*e.g.*, spare parts, spares transportation, spares storage) and maximizing operational availability. The approach uses a Monte Carlo simulation to generate scenarios based on a dataset which includes failures of the components and their associated times of failure. The objective functions are computed at each genetic algorithm generation based on all generated scenarios. Examples are given on a realistic dataset in order to illustrate the type of trade-off analyses that can be carried out. Finally, further work on a large real (not just realistic) dataset would need to be carried out for DND to benefit fully from the research.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (Technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus. e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

Level of Repair Analysis
Military Logistics
Multi-objective Genetic Algorithms
Spare Provisioning