Scenario-based Multi-Period Program Optimization for Capability Based Planning using Evolutionary Algorithms

Kamran Shafi, Saber Elsayed, Ruhul Sarker and Michael Ryan

School of Engineering and IT (SEIT), University of New South Wales Canberra,Canberra, ACT, 2600, Australia

Abstract

Capability based planning (CBP) is a strategy focused planning framework that facilitates organizations to systematically develop capacity to achieve their business objectives in highly uncertain, dynamic and competitive environments. Capability programming is an integral part of CBP which requires selecting a portfolio of capability projects for execution, referred as a capability program, such that the overall strategic risk facing the planning organization across a number of projected future operating scenarios is minimized while maintaining the most economical choice. It is a challenging optimization problem that requires handling a number of dynamic constraints and objectives that vary throughout the entire planning horizon. An optimizing simulation approach is presented in this paper that combines an evolutionary multi-objective optimization algorithm with a reinforcement learning technique to generate capability programs which optimize strategic risks and program costs across multiple planning scenarios as well as over a rolling planning horizon. The role of the optimization algorithm in this approach is to search for the non-dominated capability programs at each decision point by minimizing the strategic risks associated with individual capability projects across a number of planning scenarios as well as the total cost of the program. The reinforcement learning algorithm, on the other hand, searches horizontally within the set of non-dominated programs to minimize capability risks and costs over the entire planning horizon. The methodology is evaluated on a test problem generated based on the data distributions in an Australian Defence Capability Plan and the performance is compared with two myopic heuristic methods.

Keywords: Capability Based Planning; Program Optimization; Project Portfolio Optimization; Multi-objective evolutionary optimization.

Email address: {k.shafi, s.elsayed, r.sarker, m.ryan}@adfa.edu.au (Kamran Shafi, Saber Elsayed, Ruhul Sarker and Michael Ryan)

1. Introduction

Capability-Based-Planning (CBP) is a relatively new paradigm that provides an analytical framework for strategic or long range planning using the concept of capabilities. A capability is commonly defined as *the power or ability to do something* [40]. Although the concept of capability has various applications in diverse domains including management science, social science and engineering domains, the origins of CBP are often traced back to defence [16]. In defence jargon, capability refers to *the capacity or ability to achieve an operational effect* [17] and CBP process is defined as *an overarching framework for planning under uncertainty to provide capabilities suitable for a wide range of modern-day challenges and circumstances while working within an economic framework that necessitates choice* [16]. Currently several defence forces around the globe, including the USA, UK, Canadian and Australian Defence Force, are using CBP with some variations in the development of future force structures which can operate effectively in a diverse range of future scenarios [23].

Program management, and in specific project portfolio selection, is a key process in CBP that involves selecting a portfolio of capability projects for programming or execution over the planning horizon. Program management is an emerging area of research in strategic management discipline that aims at bridging the gap between strategy and its implementation in large organizations by focussing on managing a cohort of projects on top of traditional project management. A program is referred to as "a temporary, flexible organization created to coordinate, direct and oversee the implementation of a set of related projects and activities in order to deliver outcomes and benefits related to the organization's strategic objectives" [9]. Program management involves four key processes [34] including identification of all current, proposed and on-hold projects; prioritization or ranking of projects related to their strategic importance; selection of optimal project mix; and progress monitoring. The project portfolio selection, therefore, is a key function in program management that involves selecting a subset of desired projects to be programmed or executed that would maximize the fulfilment of the strategic objectives within the available resources. The common considerations in meeting these objectives in portfolio selection problem include budget constraints to account for project costs and the available budget; dependency constraints between the projects, positioning considerations including both the temporal and spatial constraints and threshold constraints ensuring the minimum levels of objectives are fulfilled [35].

Although project portfolio selection problem has seen increasing interest over the last decade [39], the research on this problem in the CBP context is rather scant. Some of the challenges that the CBP process in defence brings to the standard project portfolio selection problem encountered in other industries include very high cost investment decisions (typically 100s of millions of dollars), long time frames for capability development (typically 10s of years) and government imposed hard budget constraints. Among others the two key factors that differentiate defence problems from standard indus-

try problems are the type of optimization objectives and the type of uncertainty associated with these objectives. Business organisations are generally concerned with maximizing profits, revenues, market shares etc. However in defence the major concern is national security. Subsequently, minimizing strategic risk becomes the number one priority in defence portfolio optimization problems. However, unlike business organisations, strategic risk in defence is often associated with deep uncertainty which comes through a number of factors not relevant to general business problems such as changes in government and political landscapes, national security policies and priorities, local and global threat scenarios, etc. Dealing with this deep uncertainty requires use of different methodologies than the standard probabilistic risk measuring methods. Scenario-based approaches are usually used to plan under such circumstances [8] [47]. This in turn requires portfolios to be selected such that multiple conflicting objectives, e.g. minimization of strategic risks and cost and maximization of balance of investment, are traded off across a number of future scenarios. The process is often continuous since the planning landscape characterized by political, economic, social and technological dimensions change continuously. New capability projects are added regularly to the desired list and existing projects are removed from the plan due to emerging needs under changing strategic environment and political guidance. This requires that the capability program be optimized dynamically over the whole planning horizon.

The work presented in this paper is specifically motivated by the project programming problem in the defence capability planning context, in particular the program management in Australian Defence Force's (ADF) capability development process. The Defence Capability Plan (DCP) [1] outlines ADF's long-term capital program which aims at achieving the long-term strategic goals set out in the Defence White Paper (DWP). Both these documents continuously evolve to address the dynamic nature of Australia's strategic risk environment. Generally, the DWP is revised every four years. The current DWP [2] aims at making the ADF a balanced force capable of meeting every contingency in the next two decades. Various changes can trigger the DCP update process including changes made to Defence portfolio in the budget, emergence of an acute capability requirement and delays in the project schedules. The DCP is generally updated annually and contains major equipment project proposals covering a range of Defence capabilities to be evaluated by the Government over the next decade. The current DCP 2012 (public version) contains 111 projects worth $153 billion [1].

Defence capability planning is a high-stake complex iterative process which requires satisfying several competing objectives under the given constraints. A DCP needs to deliver nation's strategic goals set out in the DWP but within strict financial constraints. Other constraints include delivery schedules, the capacity of Defence and industry to deliver capability, interdependency between projects, and changing political landscapes. Figure 1 depicts the different factors and drivers considered in the formation of a DCP. Defence often has to prioritize by evaluating capabilities across several future strategic scenarios

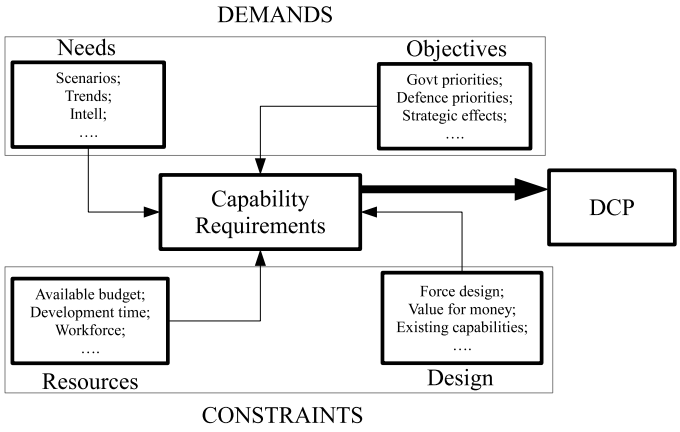

Figure 1: Defence Capability Planning

and weighing each capability on different measures of effectiveness, utility and performance. These complexities make the development of DCP an interesting optimization problem. Furthermore, the combinatorics involved in evaluating the feasibility and relative costs and benefits of a large number of DCP options necessitates the use of computational tools.

Current DCP program management practice involves ranking all capability proposals in the Defence portfolio based on a collated score assigned to each capability by Capability Managers (CM) and Subject Matter Experts (SME) across a range of strategic scenarios and then selecting the set of capabilities from the top of the list that could feasibly be programmed within the given budget constraints. Over-programming has been a common practice in previous DCPs, in order to hedge against possible project failure risks.

One limitation of this linear approach is that the decision makers do not get to see many program options to choose from and hence feel pressure to score the individual capability proposals cautiously. In addition there is no transparent mechanism available to analyze multi-dimensional risks associated with different programming options.

This paper presents a methodology for DCP program optimization under periodically changing multiple planning scenarios. The primary aim of the methodology is to search capability portfolios that lie on the efficient frontier, when traded off on a number of objectives including the cost and effectiveness scores in different planning scenarios, not only at a single time period but over the whole planning horizon. To achieve this outcome an evolutionary multi-objective optimization (EMO) approach is first used to model a single period DCP programming problem. The goal of EMO is to generate sets of non-dominated portfolios when measured against cost and effectiveness criteria across multiple planning scenarios at every time period. Evolutionary multi-objective optimization algorithms (MOEAs) are population based tech-

niques that use Darwinian evolution principles, such as natural selection and reproduction, to search the solution space. In effect, they solve optimization problems using multi-point parallel search of the solution space and therefore are considered less susceptible to local optima. In addition, they evolve a set of non-dominated solutions taking into account multiple conflicting objectives. From the DCP programming perspective this means providing the decision makers a set of non-dominated DCP programming options to choose from that are evaluated on multiple risk or objective fronts across multiple strategic scenarios.

Further, to extend the optimization problem over the entire planning horizon, we adopt an optimizing simulation approach that integrates a reinforcement learning algorithm with the MOEA to choose programs that are not only non-dominated at each time period but also perform well in the long run. This is done by adopting Q-learning [52], a well-known reinforcement learning algorithm, that works by iterating through the sets of non-dominated programs generated by the MOEA at each time period, estimating the Q-values of these programs with respect to their performance in the long run (i.e., over the entire planning horizon) over a number of simulation runs and finally allows selecting the best combination of portfolios over the whole planning horizon based on the highest estimated Q-values. Such an approach not only allows for accounting the uncertainties associated with different problem variables but is also more transparent, as it allows better analysis of individual solutions, in comparison to, for example, a monolithic approach where a combined objective function is used to model all time periods simultaneously and stochastic variables to model uncertainties.

The DCP program optimization problem discussed in this paper is closely related to the project portfolio selection or optimization problem. Some operations research (OR) techniques to model these problems date back to as early as 1970s [37], but a renewed interest could be seen towards the late 90s [4]. The application of evolutionary multi-objective and many-objective optimization algorithms to project portfolio optimization problems, however, is relatively a recent area of research [26] [11] [54]. Many large organizations have been using enterprise-wide portfolio optimization techniques for a long time. However, there is limited research in the application of these techniques to the defence portfolio optimization [28] [3] which requires dealing with specific problem characteristics discussed above.

The rest of this paper is organized as follows: the extant literature related to the problem addressed in this paper and the proposed methodology is discussed in Section 2; Section 3 presents a detailed and formal description of the capability program optimization problem; the proposed solution to address this problem is discussed in Section 4; Section 5 provides the details of our experimental setup to assess the proposed solution approach in a test DCP problem and presents and discusses the outcome of the experiments; conclusions and future work are discussed finally in Section 6.

2. Related Work

In general, portfolio optimization is a well-studied area, especially in finance and economics fields [41]. A number of techniques, including EA based approaches, have been applied to different variations of financial portfolio optimization which requires choosing a set of financial assets to maximize returns over time. For example, in his classic work Markowitz [38] studied the portfolio selection of investments in securities to minimize the risk of losses, in the form of mean-variance, and maximise the profit while considering several constraints such as cardinality and turnover constraints. Dantzig [15] solved a multi-period asset allocation portfolio optimization problem using multi-stage stochastic linear programs. Consigli and Dempster [14] proposed a CALM model, which was designed to deal with uncertainty affecting both assets (belonging to either a securities portfolio or in a market environment) and liabilities (in the form of scenario dependent payments or borrowing costs). A set of 10-stage portfolio problems were solved and discussed using different solution methods and libraries. In [10], a dynamic portfolio management problem is studied over a finite horizon with transaction costs and a risk objective function. The authors assumed that the uncertainty faced by the investor could be approximated using discrete probability distributions via a scenario approach. As a consequence, a scenario decomposition approach was used to solve the problem. Yang [53] applied a genetic algorithm (GA) based technique to solve the dynamic portfolio optimization problem with a variety of economic uncertainties. A recent survey on the use of MOEAs for portfolio optimization in finance and economic applications can be found in [41].

Various methodologies have been proposed to approach the project portfolio selection or optimization problem [4] – the main concern in this paper – especially in the area of research and development (R&D) resource allocation problems [12]. Iamratanakul et al. [29] proposed a taxonomy of project portfolio selection approaches and categorized them into six classes that include benefit measurement methods that use normative approaches such as analytical hierarchy process (AHP) and scoring models to rank projects on multiple criteria; mathematical programming approaches; cognitive emulation approaches such as decision tree and regression models; and simulation and heuristics approaches such as Monte Carlo or system dynamics based simulations; real option analysis approaches and ad-hoc modeling approaches. Using this taxonomy our work could be categorized to fall under the simulation and heuristic based approaches. Our work specifically deals with project portfolio selection problem in defence domain based on multiple objectives evaluated under multiple planning scenarios that evolve over time. Below we survey some representative examples of the work closest to our undertaken problem.

Gutjahr et al. [26] applied a bi-level modeling approach to an R&D project selection problem that includes dealing with portfolio selection as a multi-objective optimization task and the resource allocation (in particular, employee assignment) to individual projects as a scheduling task. The increase in employee competence over the planning horizon is taken into consideration and

modeled in the objective functions. Two MOEAs, NSGA-II and P-ACO, are used to solve the multi-objective portfolio selection problem. While the project costs and employee competencies are modeled as random variables, the individual test instances are generated by sampling from these distributions. Damghani et al. [32] proposed a framework for solving multi-period project selection problem in capital investment domain. Their technique is based on TOPSIS framework [13], that allows reducing a multiple objective problem to a bi-objective problem, and an extended epsilon-constraint method, which is used to generate solutions on a restricted Pareto front of the transformed bi-objective space. While four objectives related to profit maximization and cost minimization are considered in their formulation, the objective values for all projects are assumed to be fixed and predetermined.

Liesiö and Salo [36] used a scenario-based approach to model uncertainty involved in the selection of a portfolio of investment projects. The two key features of their approach include the use of a set inclusion technique to model incomplete information associated with planning scenarios and an integer programming technique to determine non-dominance relation between portfolios. Although this approach consider choosing project portfolios under multiple scenarios, the scenarios are considered as single shot pictures of distant future. Rafiee and Kianfar [43] used a multi-stage stochastic programming approach to model a multi-period project selection problem in pharmaceutical industry under multiple scenarios. However, the scenarios in their model correspond to random samples of exogenous conditions as well as probabilities associated with project success outcomes. A real-option valuation method is then used to reduce the scenario space. Moreover, their optimization model does not consider multiple objectives.

Kangaspunta et al. [31] proposed a portfolio approach to analyze the cost and efficiency of weapon systems. Their approach uses combat simulations to assess the performance of a set of military capabilities in a range of combat scenarios. Finally, an algorithmic heuristic is used to choose the systems lying on the efficient frontier which relies on enumeration of all portfolios, sorting individual systems based on a given cost criteria and then making pair-wise comparisons between feasible and *potentially* cost-efficient portfolios.

A review of these and other approaches suggest that there is an increasing interest in the use of multi-objective optimization algorithms for the portfolio selection problem for long range planning. The work presented in this paper complements these approaches and contributes both from an application as well as methodology perspective. From an application perspective, we have investigated a defence capability portfolio optimization problem which has not received much attention despite its importance. One reason for this scarcity is that the complexity of the real problem makes it hard to comprehend and scope it as an optimization problem to which popular methods, including evolutionary optimization methods, can be applied. We have, therefore, contributed to the field by providing one possible clear definition of this optimization problem, in addition to proposing a methodology to solve it. Our solution methodology combines MOEAs and reinforcement learning in an optimizing simula-

tion framework to provide an evolutionary multi-objective, multi-period and multi-scenario optimization approach. The existing MOEA based approaches for dynamic, multi-period, multi-objective problems generally rely on modeling through monolithic objective functions with stochastic variables. By integrating reinforcement learning with MOEAs our approach allows a much simpler and elegant formulation that in turns allow better management of problem complexity as well as a more transparent problem-solution analysis mechanism. Our approach allows for scaling to any number of scenarios by considering program risks in these scenarios as independent objectives. Such a problem is another good practical example for many-objective optimization research which is often criticized for lack of relevant real world problems [45]. It is anticipated that the presented work will further foster the research in long range portfolio optimization.

3. Capability Program Optimization Problem

The optimization problem considered in this paper is primarily motivated by capability planning in the defence sector, in particular the ADF. However, the characteristics of the problem are common across a number of industries, such as pharmaceutical [24] and oil and gas production sectors [48], and hence the formulation can easily be extended to cater for the specific planning requirements.

Capability planning in Defence follows a rigorous development life cycle process that starts with the identification of future needs leading to a list of capabilities (e.g. Ground Based Air and Missile Defence, Maritime Patrol and Response Aircraft, Underwater Tracking Range, etc.), belonging to a number of categories (e.g. Land, Maritime, Air, Cyber, etc.)., that would enable ADF to achieve its strategic objectives including national security, defending Australia's interest in the region and international engagement requirements. While it is desirable to acquire all the capabilities in this *wish list*, the resources needed to acquire these capabilities always exceed the available resources. Despite their importance therefore, a limited number of capability projects can be scheduled to be programmed at any given point in time. A second challenge in selecting amongst the desired capabilities is the deep uncertainty associated with their future use. Most capabilities usually have long lead times with typical development life cycles of 20 to 30 years. Given the pace of current technological developments predicting future requirements is a next to impossible task. On the other hand, any decision made about acquiring a capability is almost irreversible and carries a risk of huge opportunity loss due to the high cost involved in the capability development. A number of representative future scenarios are therefore used to carefully evaluate the potential use of these capabilities. In fact, it is safe to assume that the exhaustive *wish list* is generated by trying to cover potential risk in all of the representative scenarios.

Hence, the key objective of capability program optimization is to select a set of capability projects to acquire or program such that the effectiveness of these projects is maximized across a number of potential future scenarios and

the cost to program these projects is minimized. In addition the problem is continuous; i.e., the scenarios used for evaluating capability needs evolve with time, new capability needs emerge and irrelevance of existing capabilities is exposed. Since decisions about acquiring capabilities are made at present, these future changes are needed to be taken into account when programming capability projects.

Given this context, we break down this problem into two sub tasks that simplifies the analysis as well as support developing the solution methodology.

3.1. Single-period Formulation

The first sub task is to model the single period optimization problem. This is done by ignoring the temporal dimension and focusing only on the optimization of multiple objectives at a single time step in the planning horizon. Let $x_{ij} \in X$ be the i_{th} capability project belonging to the j_{th} category in the current wish list of the desired capabilities X with a given development cost c_{ij}. Further assume a number of planning scenarios under which the effectiveness of each of the capabilities in X is evaluated. Notice that the definition of scenarios in our formulation is notional. A typical scenario in defence may refer to a targeted operation in a specific physical terrain, threat environment and a predicted enemy strength. In such a scenario specific capabilities, e.g. unmanned ground or air vehicles, would vary in their effectiveness from another scenario which differs from the first in the above three features. Hence, we assume that a capability in every scenario performs differently on risk or effectiveness scores. In other words, each capability has a set of effectiveness scores corresponding to the number of scenarios. Conversely, a scenario is defined by the set of effectiveness scores for all capabilities in the wish list. However, this assumption is not restrictive and does not prevent our methodology being used with the real planning scenarios, so long as a mechanism of allocating risk or effectiveness scores to capability projects in these scenarios could be defined. Let r_{ij}^k be a value by which capability x_{ij} reduces the amount of risk[1] in scenario k. Also notice that while we assume risk or effectiveness scores for capability projects to vary across planning scenarios, project costs remain constant and independent of scenarios. This assumption is not too unreasonable given that capability projects do not vary with scenarios.

Using a binary notation, our decision variable x_{ij} can be defined as:

$$x_{ij} \in X = \begin{cases} 1 & \text{if capability } x_{ij} \text{ is selected to be programmed.} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

A single period multi-objective program optimization problem is then to select a set of capability projects (or a portfolio P) to be programmed in the

[1]For discussion in this paper, risk reduction and effectiveness are used interchangeably. The more a capability is able to reduce a risk, the more effective it is in that scenario. The optimization problem can be posed as either minimization of strategic risk or maximization of effectiveness.

current time period that minimizes (1) the cost and (2) the risk (or maximizes the effectiveness) across all planning scenarios, over the space of all feasible portfolios. Formally using the above notation, the two objectives are given in Equations 2 and 3:

$$\min \sum_i \sum_j c_{ij} x_{ij} \tag{2}$$

and

$$\min \sum_i \sum_j r_{ij}^k x_{ij}, \quad \forall k \tag{3}$$

subject to:

$$\sum_i c_{ij} x_{ij} \leq B_j(1+\delta_j), \quad \forall j \tag{4}$$

$$\sum_i \sum_j c_{ij} x_{ij} \leq \sum_j [B_j(1+\delta_j)] \tag{5}$$

$$0 \leq \sum_i \sum_j r_{ij}^k x_{ij} \leq 1, \quad \forall k \tag{6}$$

$$x_{ij} \in \{0,1\} \tag{7}$$

Here $k \in [1, K]$ denotes a scenario and K the total number of scenarios and hence the second objective is a composite function of K objectives. B_j in Equations 4 and 5 refers to the given budget for a category j and δ_j is a violation threshold set up as a small percentage (< 0.1) of the given budget. The two equations (4 and 5) simply constrain the total cost of a program to the available budget plus a threshold in each category as well as the total budget. Note that satisfying constraint in Equation 4 will implicitly satisfy Equation 5. However, reverse may not be true. Hence, we needed to keep both constraints. (Same is true for constraints listed in Equations 13 and Equations 14 discussed in next section.) Equation 6, on the other hand, constrains the portfolio risk values to a normalized risk (or effectiveness) score within [0,1] interval. Here a risk score of 0 (or effectiveness score of 1) implies that the set of programmed capabilities could potentially cover all perceived or known risks. A risk score less than 0 (or effectiveness score greater than 1) may imply over protection which in turn implies that some of the capabilities could be removed from such a portfolio without compromising the risk in a given scenario. Conversely, a risk score of 1 (or effectiveness score of 0) implies effectively no protection against known risks. Practically, such a constraint would be ineffective if the sum of risk or effectiveness scores for all capabilities in a wish list are bounded not to exceed 0% risk (or 100% effectiveness). The data generated for test problems used in this study follow this latter approach (see Section 5 for details).

3.2. Multi-period Formulation

Recall from our previous discussion that the capability program comprises a set of projects or a portfolio to be programmed over the entire planning horizon (10 years for the considered DCP problem). The planning problem therefore requires considering emerging requirements, changes in the environment and hence their effect on planning scenarios, capturing variations in available resources such as budget, and so on. Extending the above multi-objective formulation to multiple time periods, therefore, implies selecting a capability program that would minimize the above formulated cost and risk objectives over the whole planning horizon. Formally, the decision variable over time x_{ijt} is defined in Equation 8:

$$x_{ijt} \in X_t = \begin{cases} 1 & \text{if capability } x_{ij} \text{ is selected to be programmed at time } t. \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

and the above objective functions over time become:

$$\min \sum_i \sum_j c_{ijt} x_{ijt}, \quad \forall t \tag{9}$$

and

$$\min \sum_i \sum_j r^k_{ijt} x_{ijt}, \quad \forall k, \forall t \tag{10}$$

subject to:

$$\sum_i c_{ijt} x_{ijt} \leq B_{jt}(1 + \delta_j), \quad \forall j, \forall t \tag{11}$$

$$\sum_i \sum_j c_{ijt} x_{ijt} \leq \sum_j [B_{jt}(1 + \delta_j)], \quad \forall t \tag{12}$$

$$0 \leq \sum_i \sum_j r^k_{ijt} x_{ijt} \leq 1, \quad \forall k, \forall t \tag{13}$$

$$x_{ijt} \in \{0, 1\} \tag{14}$$

In simple words the above formulation assumes that we are solving a different problem (or more correctly different instances of the same problem introduced by changes in scenarios) at each time period. That is, we assume that we do not have any information about the future, neither can we estimate expectations due to deep uncertainty, which would require using random variables in the formulation as is done in stochastic programming [**?**] based models where the coefficients and right-hand sides are random variables. In our case, the formulated deterministic models can deal with uncertainty in scenarios through the use of RL and multiobjectivity through MOEAs. This allows us to deal with uncertainty associated with future decisions while dynamically

optimizing over a constraint search space defined by the set of Pareto or non-dominated portfolios available at each time step in the planning horizon. The complete solution approach is detailed in Section 4.2.

3.3. Model Assumptions

A number of simplifying assumptions are made in the above formulation. Specifically, it is assumed that:

- The wishlist of all projects to be programmed over the entire planning horizon is assumed to be known at present. However, our model allows addition of a small proportion of new projects (emerging requirements) at every future time period. Furthermore, the available budget and the effectiveness scores for each project in different scenarios are considered unknown and changing over time.
- Each project has a fixed completion period.
- All projects have equal priority.
- There are no interdependence between projects, i.e. all projects can run in parallel.
- Cost of each project is known and is equally distributed over the entire planning periods. The cost is also considered fixed across the planning scenarios.
- Once a project is programmed it cannot be removed from the project list until completion.

Although some of the assumptions are not representative of the actual capability programming problem, specifically project priorities and their interdependence, the problem is already quite complicated without the addition of such constraints. We hypothesize that most of these assumptions can be relaxed through additional feasibility constraints to the formulation without the loss of generality. However, exploration of this hypothesis is left for future work.

4. Solution Approach

The program optimization problem formulated in Section 3 presents two key optimization challenges related to two sub tasks mentioned above. The solution approach presented in this section is developed to address the two tasks using an integrated framework which combines evolutionary multi-objective optimization and an RL method in a simulation setting.

4.1. Single-period Multi-Objective Optimization

The first optimization challenge relates to finding a set of non-dominated portfolios, consisting of capability projects, that minimize the risk (or maximize effectiveness) across a number of planning scenarios as well as the cost to fund them, which is constant across all different scenarios. This could potentially be formulated as a bi-objective optimization task where cost is treated as one objective and an aggregated risk score over all scenarios as the other. The risk score aggregation could, for instance, follow a weighted approach. However, one problem with such an approach is that it does not allow a transparent comparison between different risk dimensions and depends on weight allocations by experts. An alternative approach could be to formulate it as K bi-objective optimization problems[2]. Here K corresponds to the number of planning scenarios used for evaluating the effectiveness of each capability and the two objectives correspond to the total cost and the total effectiveness of capabilities in a given portfolio under a given scenario. However, the problem with this approach is that it does not provide a seamless way to reconcile between the set of Pareto optimal solutions obtained through solving the K bi-objective optimization problems and compare them on a common axes. To overcome this challenge, we instead model this problem, in this paper, as a multi-objective optimization problem with $K + 1$ objectives, where K objectives correspond to the total effectiveness scores of a given portfolio in each of the K planning scenarios and the additional objective corresponds to the cost of the portfolio, which we assume fixed across all scenarios.

An MOEA, in specific MOEA/D [55], is utilized in this paper as an approach to address this multi-objective optimization problem. MOEAs are metaheuristic search techniques which have shown competence in finding globally optimum solutions in large search spaces. MOEA/D is one of the recent state-of-the-art MOEA which solves a multi-objective optimization problem by decomposing it into multiple scalar optimization problems and then apply an EA (here differential evolution (DE) algorithm is considered [21, 20, 5]) to solve all single objective optimization problems simultaneously. The details on MOEA/D algorithm are provided in Appendix 7.1. Since DE is designed to work with real-valued vectors, and a binary formulation is used here to represent solutions (see Equation 1), a real to binary encoding is used as follows: First a dummy array of real values within [0,1] is initialized, along with a binary array representing all projects. Then, simple DE mutation and crossover operators are used, where for each individual ($\overrightarrow{x_z}$) in the dummy array a new child ($\overrightarrow{u_z}$) is generated as follows:

$$u'_{z,j} = \begin{cases} x_{a_1,j} + F.(x_{a_2,j} - x_{a_3,j}) & if(rand \leq cr \text{ or } j = j_{rand}) \\ x_{z,j} & \text{otherwise} \end{cases}$$

[2]Or K multi-objective optimization problems, if additional objectives, such as balance of force-mix, are considered.

where $rand\,[0,1]$, a_1, a_2 and a_3 are random integer numbers $\in$ [1,PS] and j_{rand} 1,2,D is a randomly selected index, which ensures $\overrightarrow{U_z}$ gets at least one component from $\overrightarrow{x_z}$ Consequently, for the main 0-1 array, the new individual is set as follows:

$$u_{z,j} = \begin{cases} 0 & if(u'_{z,j} \leq 0.5) \\ 1 & \text{otherwise} \end{cases}$$

if the new child is better than its parent, both ($\overrightarrow{u_z}$) and ($\overrightarrow{u_z}$) are copied to the next generation.

A new breed of evolutionary algorithms has emerged to scale MOEAs to problems with larger number of objective functions, known as many-objective evolutionary optimization algorithms. The leading algorithms in this area have been shown to scale up to several tens of objectives [7] [18]. However, since in this paper we only consider a maximum of four planning scenarios and therefore five objective functions to optimize, we restrict ourselves to MOEA/D, which has shown to scale well to greater than three objectives [30] [51] in comparison to other famous MOEAs, including NSGA-II. Further, a modified version of MOEA/D [6] is used that can has shown even better handling of problems with more than three objectives. Also, the original MOEA/D was proposed for unconstrained optimization problems. Further, a modified version of MOEA/D [6] is used that can has shown even better handling of problems with more than three objectives. Also, the original MOEA/D was proposed for unconstrained optimization problems. The modified MOEA/D used in this paper relaxes this assumption and allows handling constraints. In particular, the superiority of feasible solutions method is used to handle the constraints using the following criteria: (i) a feasible solution is always preferred over an infeasible solution; (ii) a solution with higher fitness is preferred when comparing between two feasible solutions; and (iii) a solution with smaller sum of constraint violation is preferred when comparing between two infeasible solutions. The sum of constraint violation is calculated as follows:

$$vio(\overrightarrow{x}) = \sum_{k=1}^{K} max(o, g_k(\overrightarrow{x})) + \sum_{e=1}^{E} max(o, |h_e(\overrightarrow{x})| - \varepsilon) \tag{15}$$

where $g_k(\overrightarrow{x})$ is the k^{th} inequality constraints, $h_e(\overrightarrow{x})$ the e^{th} equality constraint. The equality constraints are also transformed to inequalities of the form:

$$-\varepsilon \leq h_e(\overrightarrow{x}) \leq \varepsilon \tag{16}$$

4.2. Multi-Period Optimization

The second challenge relates to extending the multi-objective optimization methodology presented above over the entire planning horizon. This is im-

portant because the decisions made at the current time need to consider the effect of future changes such as updates about available resources and capability needs.

We use an optimizing simulation approach that integrates MOEA/D and RL with a Monte Carlo simulation to address this problem. Put simply, the simulation is used to generate different state variables (which involves updating the budget, capability list and scenarios through generating effectiveness scores from a given distribution for each of the capability projects) at each time step of a simulation iteration, MOEA is used to choose actions (one of the non-dominated program options from the current solution) and the RL is used to update the Q-value for selected state-action pairs using the standard Q-learning equation [52]. During learning the actions are selected either randomly or based on the maximum Q-values to balance between explore and exploit respectively. The detailed algorithm is provided in the next section.

The overall objective of this stage of optimization is to select a set of non-dominated programs at each time period in the planning horizon so that the optimization of objectives, i.e., the minimization of the cost of programs and strategic risks across all planning scenarios, can be extended over the entire planning horizon. In other words, a non-dominated choice of projects at a single point in time might not be so in the long run. This is similar to playing a game of chess, at a given move you might need to sacrifice a piece on the board with an objective of winning at the end of the game. Or to a path planning problem where the set of non-dominated programs at each period can be considered as nodes in the network and the goal then is to choose a path consisting of a set of nodes that is better overall in terms of multiple-objectives. However, unlike pre-defined nodes in a path planning problem, the nodes here are generated based on the choice of the nodes in the previous time steps.

4.3. Putting it all together

In summary, our scenario-based multi-period multi-objective program optimization methodology involves the following optimizing simulation steps:

1. At each time step generating the list of desired capabilities which depends on the decisions made in all previous time steps and the emerging capability requirements, updated scenarios (defined by the risk scores of each project in the current capability list) and the available budget;
2. Solving a $K + 1$ objectives optimization problem where K is the number of planning scenarios across which the strategic risk is minimized in addition to the cost of the program;
3. Choosing a portfolio to program from the non-dominated set, obtained through the optimization procedure in the previous step, either randomly during exploration mode or based on the best Q-value in the exploitation mode;
4. Iterating through these steps until the stopping criteria is reached.

Algorithm 1 Multi-period Multi-Objective Program Optimization Algorithm

$m \leftarrow 0$;
$S_0 \leftarrow$ initialize current state at $t = 0$
$Q \leftarrow []$; (initialized to 0 or infinity depending upon the direction of optimization)
$P_0 \leftarrow GetParetoSet(S_0)$
while $m < M$ **do**
  $t \leftarrow 0$
  $S_t \leftarrow S_0$
  $P_t \leftarrow P_0$
  while $t < T$ **do**
    $a_t \leftarrow ChooseAction(P_t, S_t)$
    $S_{t+1} \leftarrow GetNextState(S_t, a_t)$
    $P_{t+1} \leftarrow GetParetoSet(S_{t+1})$
    $R_t \leftarrow ComputeReward(a_t, S_t, P_{t+1}, S_{t+1})$
    $Q(S_t, a_t) \leftarrow (1 - \alpha)Q(S_t, a_t) + \alpha[R_t + \gamma \max_{PS} Q(S_{t+1}, a_{t+1})]$
    $t \leftarrow t + 1$
  end while
  $m \leftarrow m + 1$
end while
return

A more detailed pseudocode listing the main functions is given in Algorithm 1 where the explanation of terms used in the algorithm is given in Table 1.

In every simulation iteration m, a project portfolio (represented by action a_t) is selected from the non-dominated set, obtained through solving a single period multi-objective optimization problem at t. Since the conditions at $t = 0$ are known, the non-dominated set obtained from the master project list at $t = 0$ is considered fixed for all simulation runs. An action is selected (the $ChooseAction$ function) either randomly or based on the highest Q-value with a given probability controlled by a parameter Δ, which provides balance between exploration and exploitation cycles. In the simplest strategy, Δ is set to 0.5 that allows uniform random choice between explore and exploit trials. Other strategies could be used to adapt Δ based on the learning outcomes. The next state (S_{t+1}) is then generated based on the selected action. We need to do this because the list of projects available in the next time step depends on the portfolios selected in the previous time steps. Note that the budget and effectiveness scores (or scenarios), as explained in Section 3.3, are not considered dependent on the selected actions in the previous time steps.

The next important step in the algorithm is the computation of reward for the selected action. This is done by first solving the single-period optimization problem for the next time step using S_{t+1} and obtaining the non-dominated set P_{t+1}. The reward R_t for (S_t, a_t) is then computed as the weighted sum of two

M	represents the number of simulation cycles and $m \in 0, 1, \cdots, M-1$
T	represents the length of the planning horizon (10 years for DCP problem) and $t \in 0, 1, \cdots, T-1$
S_t	represents a state at time t which is represented by the capability list X_t at time t, K scenarios (marked by K different effectiveness scores for each project in X_t) and the available budget B_t at time t
P_t	represents the set of non-dominated portfolios at time t
a_t	represents the portfolio selected from P_t either randomly (in exploration mode) or based on the best Q score (in exploitation mode)
Q	represents the state-action table used to store the Q scores or factors

Table 1: Explanation of terms used in **Algorithm 1**

factors as follows:

$$R_t = w_1 R_C + w_2 R_E \tag{17}$$

where R_C refers to the reward contribution due to the cost factor and is computed as (Equation 18) (i) the sum of the absolute difference between the available budget (B_t) and the cost of the selected portfolio (C_a); and (ii) the sum of the absolute differences between the cost of each portfolio in P_{t+1} (C_i) and the budget in the next time step B_{t+1}. Both terms are normalized using the given budget values and the number of portfolios in respective non-dominated sets (N_P).

$$R_C = \frac{1}{2}\left[\frac{(|C_a - B_t|)}{B_t} + \frac{\sum_{P_{t+1}}(|C_i - B_{t+1}|)}{N_P B_{t+1}}\right] \forall i \in P_{t+1}. \tag{18}$$

The premise of using such a formulation is to provide higher rewards to the actions (or portfolios) whose costs are as close as possible to the reference point, i.e. the available budgets so that budget over or underspending could be minimized. Moreover, the second term in the expression aims at capturing the effect of action choice on the generation of future options (i.e., the generation of non-dominated set in the following time step). Notice that this approach allows us to compute portfolio values in forward direction, unlike the more complicated classical dynamic programming approach which relies on updating the value functions recursively.

R_E, on the other hand, refers to the reward contribution due to the second composite objective i.e. the portfolio effectiveness scores in multiple scenarios. It is computed similar to R_C (Equation 19) except that the additional summation is applied due to multiple scenarios and the reference value of 1

corresponds to 100% effectiveness in each scenario.

$$R_E = \frac{1}{2}\left[\frac{(\sum_{K_t}(1-E_{a_k})}{K_t} + \frac{\sum_{P_{t+1}}\sum_{K_t+1}(|1-E_{i_j}|)}{N_P K_{t+1}}\right] \forall i \in P_{t+1}|k \in K_t|j \in K_{t+1}. \tag{19}$$

Finally, w_1 and w_2 in Equation 17 are weights that moderate decision maker preferences on the two factors. In our formulation we used equal weights for both factors.

Once the reward is computed for the selected portfolio (a_t), Q-values are updated using the standard Q-learning equation (line 14 in Algorithm 1). Here $Q(S_t, a_t)$ corresponds to the Q-value of choosing action a_t in state S_t (see Table 1), $\alpha \in [0, 1]$ is the learning rate that determines the degree to which the effect of new information is taken into account; $\alpha = 0$ means no learning and $\alpha = 1$ means only the most recent updates are taken into account. Usually a small α value is chosen for stochastic environments. $\gamma \in [0, 1]$ is the discount factor that determines the degree to which long term rewards are taken into account. The reward R_t is explained above. Simply put, this equation estimates the value of taking a specific action in a specific state. Notice that the state-action representation used in our formulation may not be scalable understandably, as the problem we are dealing with is continuous. Obviously, a much more sophisticated approximation approach, such as those based on artificial neural networks or regression models [42], is needed to represent value function on some aggregation level. However, since the focus of this paper is the proof of our proposed methodology instead of its scalability, we leave this work for future exploration and in this paper stick to a simpler look up representation. Further, to make the problem tractable, we restrict the search space by keeping the state-space constant across simulation runs. This is done by allowing the capability project list, budget and effectiveness scores in scenarios to vary between time periods but not between multiple simulation runs. In other words, the state-space is sampled only once in a single experimental run. However, this limitation is somewhat compensated since the results are averaged over multiple experimental runs (Section 5). Notice also that even a single sample generates a significant number of state-action pairs and hence the search space is still quite large. A hash-table representation is used for faster storing and matching operations during this process.

The steps explained above are repeated for every update period within the planning horizon (the inner loop in Algorithm 1) and over a specified number of simulation runs (the outer loop in Algorithm 1).

5. Experimental Setup

5.1. Test Problem Generation

The program optimization problem undertaken in this work is motivated primarily by the capability programming problem in the Australian Defence

Organisation. However, the defence capability planning data is highly classified information for obvious reasons. To evaluate our methodology, in this paper, we relied on the public version of the Defence Capability Plan (DCP2012) to generate the test data. Defence releases public DCPs periodically for the industry guidance in order to prepare them to deliver forthcoming defence's capability needs. DCP provides a list of proposed major capital equipment acquisitions that are scheduled for the Australian government consideration in the next ten years. The current public DCP2012 lists 111 projects which cover a range of defence capabilities, including Maritime Forces, Land Forces, Air Forces, Strike, and Network Centric Warfare. The test problem used in this paper is generated based on the distributions built from DCP2012 data. A list of parameters used to generate the test problems are listed below:

- Planning horizon (T): is set to ten years based on the general DCP setting but can be increased or decreased to suit the planning requirements.
- Planning scenarios (K): is set to four. The actual scenario generation methodology could be quite complex and may involve input from senior decision makers, domain experts and stakeholders. In this paper, however, we use a notional scenario representation which only requires defining capability effectiveness scores as explained below. So, although the number of scenarios used in the test problem is arbitrary, the number can be trivially increased or decreased.
- The number of capability projects at time $t = 1$ ($P_{t=1}$): is set to 60 projects and then increase by 10 projects at each subsequent year, i.e., $P_{t=10} = 150$. The range is based on the number of projects generally get listed in DCPs and can be varied.
- The number of different capability categories (J): is set to five based on the number of categories in DCP2012.
- Annual budget (B_t): is generated based on a probability distribution derived from DCP spending over the forward budget estimate period of four years in different project categories.
- Cost of capability i belonging to category j (c_{ij}): is generated based on a probability distribution derived from the projects cost provided in DCP2012 (See Figure 2). The capability costs generally follow a power-law distribution, which signifies a few projects with large budgets and majority projects with relatively smaller budgets. This makes the optimization problem even harder due to the limited flexibility in slipping projects.
- Effectiveness of x_{ij} in scenario k (r_{ij^k}): is generated based on a probability distribution derived from industry readiness scores given in DCP2012 (See Figure 3.). The distribution for generating effectiveness scores for multiple scenarios are then derived by randomly perturbing the mean and the variance of the probability distribution derived from the above

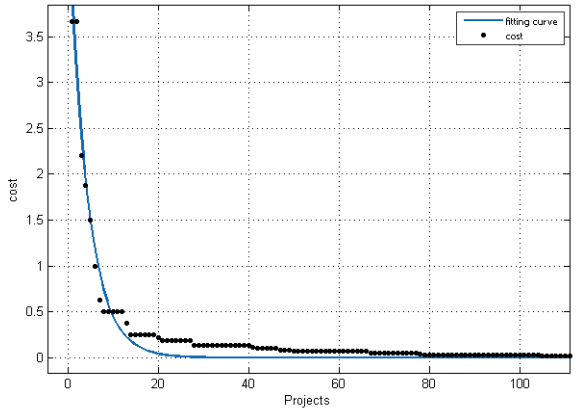

Figure 2: Cost fitting curve

method. Determining capability effectiveness is a planning scenario is a non-trivial task and a major research challenge in defence research [50]. However, since in this paper our focus is on the optimization methodology, we use this crude estimation technique to generate effectiveness scores. Note that changing the generation function for effectiveness values, or any of the above parameters for that matter, does not change the formulation of the problem or our optimization methodology.

5.2. Baseline Heuristics

We implemented two baseline heuristics to solve the DCP optimization problem in order to provide an objective comparison of the performance of the proposed algorithm.

Both heuristics use a myopic policy. The first heuristic uses a random approach to form portfolios of projects within the constraint boundaries. At time $t = 0$ the current list of projects is initialized randomly. Then each project in the list is uniform randomly selected to the current portfolio as long as the updated portfolio does not violate any feasibility constraints, including the budget constraints within a threshold. The list is scanned until the budget threshold is met. In the following time step the project list is updated with new information based on the selected portfolio in the previous time step and the portfolio is updated using the above procedure. The process is repeated for the whole planning horizon to get the final solution.

The second heuristic follows a similar procedure. However, instead of a random selection at each time step, the second heuristic selects projects greedily from the current list to form a portfolio. This is done by computing the average effectiveness scores for each project in the list over all scenarios. The project list is then sorted from highest to lowest effectiveness scores and a port-

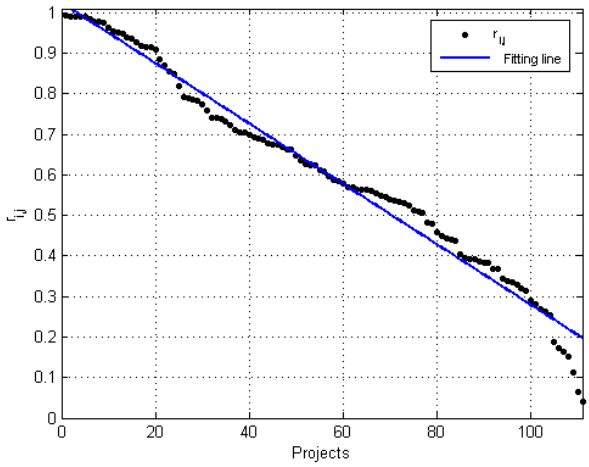

Figure 3: Effectiveness fitting curve

folio is formed by selecting the projects from the top of the list within the given budget constraints.

5.3. Results and Discussions

The experimental results are based on 20 independent runs of the proposed algorithm (see Algorithm 1), where each run consisted of 500 simulation cycles (M). To reiterate, each simulation run consisted of solving a multi-objective optimization problem for each of the 10 planning years iteratively taking into account the decisions made in the previous time steps. Each run of the multi-objective optimization algorithm (modified MOEA/D) in turn consisted of evaluating 1000 individuals or solutions (population size) for 30 generations. Hence each simulation run consisted of $30 \times 1000 \times 10 = 300,000$ objective function evaluations. In other important parameter settings, crossover rate C_r was set to 0.95 whereas the scaling factor F was chosen uniform randomly from the range $[0.4 - 0.95]$. For a fair comparison, the results of the baseline heuristic algorithm were also averaged over 20 independent runs.

Figure 4 shows the effectiveness values for selected solutions averaged over all four scenarios obtained by the three algorithms over ten years. The y-axis range from 0 to 1 respectively corresponding to 0% effectiveness (or no risk coverage) to 100% effectiveness (or complete risk coverage) averaged over all scenarios. Two curves are presented for each of the three algorithms corresponding to the best and worst effectiveness results obtained out of all simulation cycles in 20 independent runs, respectively. To elucidate further, the best here corresponds to path that achieved the highest Q-value in 500 simulation cycles over all 10 years. On the other hand, the plots of worst scores correspond to the poorest solution of the best 20 runs. In other words, the best and worst

scores correspond to the best of the best and worst of the best in 20 runs, respectively. For a better resolution, Figure 5 depicts the same results as Figure 4 but only for average best scores corresponds to the average of 20 best solutions achieved by all three algorithms.

As can be observed, these results depict that the proposed optimizing simulation algorithm overall performs better than the baseline heuristics algorithm in terms of achieving solutions which have higher effectiveness scores averaged across all scenarios. Specifically, the algorithm provides much improved performance than the baseline algorithms between years 1 and 4. Disappointingly, the improved performance does not seem to be asymptotic over the whole planning horizon as the difference between the effectiveness scores narrows in the middle years (5 to 10). Also, the performance of all the algorithms follows a downward trend between years 5 and 7, relative to their respective performances over other years. We presume, rather strongly, that this is more of an artifact of the synthetic data used in these experiments. The optimization algorithms could struggle, for instance, if the number of feasible solutions is very limited and there is no or hardly any room for improvements. In such cases, a random solution would be as good as the outcome of any optimization algorithm. However, in real problems one would expect a much greater flexibility over the range of solutions and hence the need for optimization. This hypothesis is further confirmed by a regained performance by both algorithms over the subsequent years (8 to 10), when the performance of the proposed optimization based algorithm starts improving over the baseline algorithm once again. Also notice in Figure 4 that the best solution found in these experiments provides the worst effectiveness score in year 7. This could be surprising but recall our definition of best here corresponds to best results overall years. Furthermore, error bars, which show the confidence intervals of data along the average effectiveness curves, demonstrate that the proposed method is able to attain better results than the baseline heuristics.

Figures 6 and 7 show the results obtained for the second objective, i.e. the cost error relative to available budget, in a similar manner. Notice that we constrained the cost of portfolios to a maximum threshold, i.e., -0.5, to limit overspending. As it is better to avoid overspending, the solution which has a cost error closer to zero is preferable. Based on this definition, the proposed algorithm considered the best.

5.4. Effect of Q-learning

In this subsection, we would like to demonstrate the benefit of using Q-learning. To do this, the algorithm was run 20 times without applying the Q-learning part. In this case, at every time period, a solution from the non-dominated set was randomly selected. Subsequently, the average effectiveness scores of 20 runs averaged over all four scenarios were calculated and then depicted in Figure 8. The results demonstrate that the algorithm with the Q-learning mechanism was consistently able to obtain better results. For a further illustration, the best solution based on that second objective was recorded, as

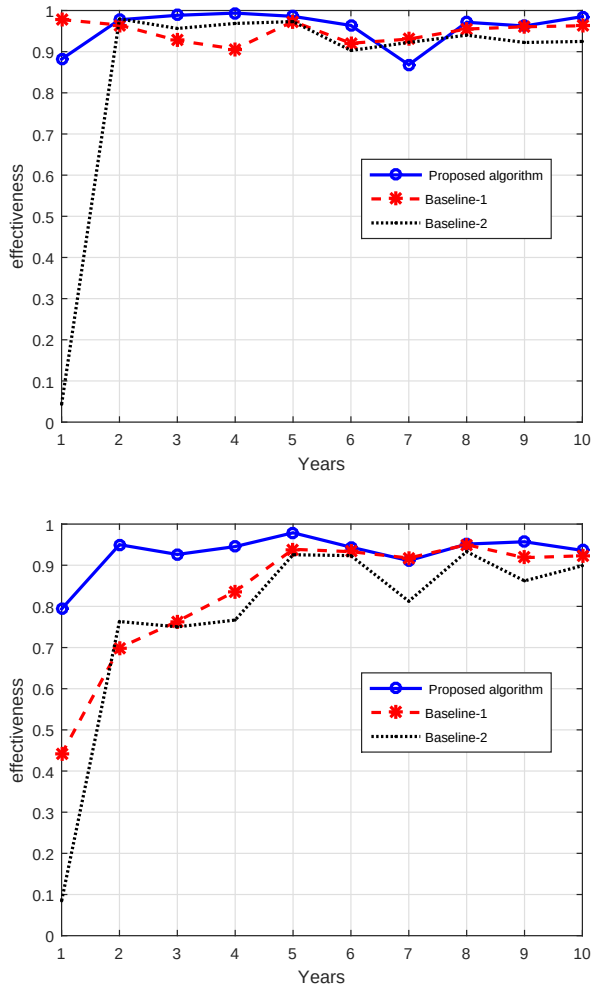

Figure 4: Best and worst effectiveness scores obtained by the proposed optimizing simulation method and the baseline heuristic over 20 runs, respectively

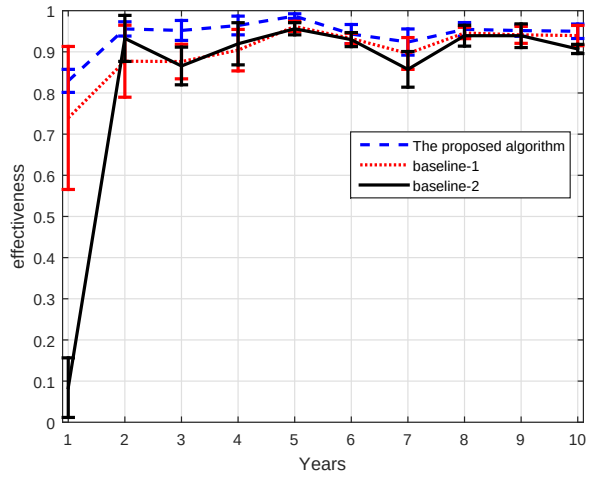

Figure 5: Average effectiveness scores and error bars in four scenarios over the planning horizon obtained by the proposed optimizing simulation method and the baseline heuristic. The results are averaged over 20 runs.

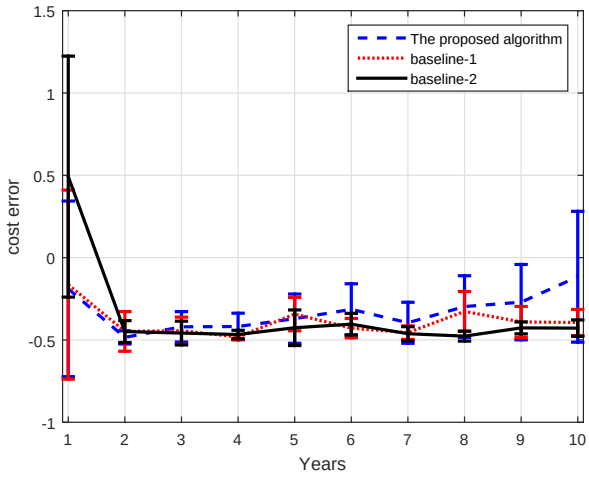

Figure 6: Average cost error and error bars over the planning horizon obtained by the proposed optimizing simulation method and baseline heuristics. The scores are averaged over all scenarios as well as over 20 runs.

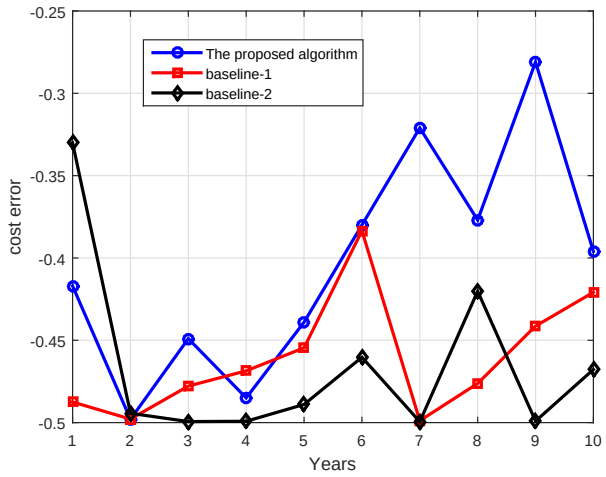

Figure 7: Best cost error over the planning horizon obtained by the proposed optimizing simulation method and the baseline heuristic. The best scores are out of all 20 runs.

shown in Figure 9. Over the 10 years, the variant with the Q-learning component was able reduce the overspending.

6. Conclusions

Capability programming under defence strategic planning is a key task that involves selecting a set of capabilities to be acquired over a given planning horizon while optimizing the financial and other resources as well as minimizing the envisaged strategic risks in a number of future planning scenarios. In this paper, we first formulated this problem as a dynamic multi-objective optimization task and then proposed an optimizing simulation methodology that combines evolutionary multi-objective optimization and a RL algorithm to select Pareto optimal capability portfolios across a number of objectives as well as over time. The methodology is evaluated on a test problem generated from public version of the ADF's 2012 DCP. The performance of the proposed technique is compared with a baseline heuristic developed using a myopic policy. Although the proposed methodology is evaluated on a defence related problem, it has much wider implications as capability programming has similarities with planning problems in various other industries including pharmaceutical, mining, oil and gas and R&D sectors.

A number of future directions stem from this work. First, we would like to extend the fidelity of the optimization model by removing simplifying assumptions and incorporating a number of real-world complexities and constraints such as capability interdependencies, priorities and project schedules. Second, the exogenous variables, such as given budget, could be modeled as stochastic processes across simulation runs to capture uncertainty associated

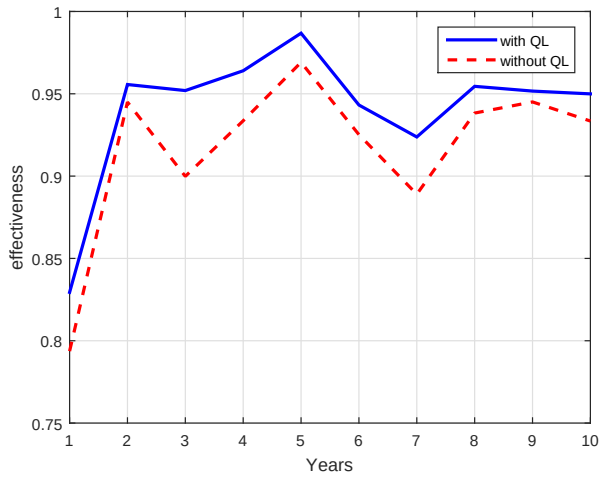

Figure 8: Average effectiveness scores of four scenarios over the planning horizon obtained obtained by the proposed algorithm with and with and without Q-learning

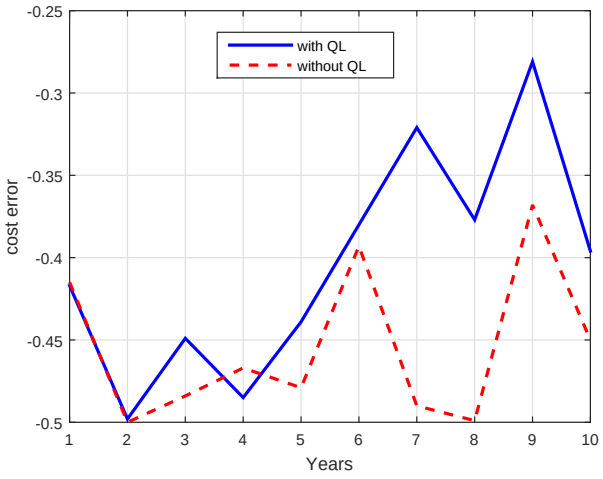

Figure 9: Best cost error over the planning horizon obtained by the proposed algorithm with and with and without Q-learning

with these variables. This would require extending the Q-learning component with advanced function approximation techniques such as neural networks. A significant subsequent challenge is to reduce the computational complexity of the current algorithms. Design of benchmark test problems capturing the essence of such problems is also an important direction that we would like to explore in future. Finally, it would be interesting to link the developed models with other related optimization problems in the organization to provide an enterprise-wide optimization framework.

References

[1] 2012, D. (2012). Defence Capability Plan 2012: Public Version.

[2] 2013, D. (2013). 2013 Defence White Paper.

[3] Adams, A., Bala, E., Minner, B., and Woodland, T. (2009). Defense portfolio analysis.

[4] Archer, N. P. and Ghasemzadeh, F. (1999). An integrated framework for project portfolio selection. *International Journal of Project Management*, 17(4):207–216.

[5] Asafuddoula, M., Ray, T., and Sarker, R. (2015). An improved self-adaptive constraint sequencing approach for constrained optimization problems. *Applied Mathematics and Computation*, 253:23–39.

[6] Asafuddoula, M., Ray, T., Sarker, R., and Alam, K. (2012). An adaptive constraint handling approach embedded moea/d. In *Evolutionary Computation (CEC), 2012 IEEE Congress on*, pages 1–8.

[7] Bader, J. and Zitzler, E. (2011). Hype: An algorithm for fast hypervolume-based many-objective optimization. *Evolutionary Computation*, 19(1):45–76.

[8] Baker, S., Bender, A., Abbass, H., and Sarker, R. (2007). A scenario-based evolutionary scheduling approach for assessing future supply chain fleet capabilities. In *Evolutionary Scheduling*, pages 485–511. Springer.

[9] Barclay, C. and Osei-Bryson, K.-M. (2009). Toward a more practical approach to evaluating programs: The multi-objective realization approach. *Project Management Journal*, 40(4):74–93.

[10] Barro, D. and Canestrelli, E. (2005). Dynamic portfolio optimization: Time decomposition using the maximum principle with a scenario approach. *European Journal of Operational Research*, 163(1):217 – 229. ¡ce:title¿Financial Modelling and Risk Management¡/ce:title¿.

[11] Carazo, A. F., Gmez, T., Molina, J., Hernndez-Daz, A. G., Guerrero, F. M., and Caballero, R. (2010). Solving a comprehensive model for multiobjective project portfolio selection. *Computers & Operations Research*, 37(4):630 – 639.

[12] Carlsson, C., Fullér, R., Heikkilä, M., and Majlender, P. (2007). A fuzzy approach to r&d project portfolio selection. *International Journal of Approximate Reasoning*, 44(2):93–105.

[13] Chen, C.-T. (2000). Extensions of the topsis for group decision-making under fuzzy environment. *Fuzzy sets and systems*, 114(1):1–9.

[14] Consigli, G. and Dempster, M. (1998). Dynamic stochastic programmingfor asset-liability management. *Annals of Operations Research*, 81(0):131–162.

[15] Dantzig, G. and Infanger, G. (1993). Multi-stage stochastic linear programs for portfolio optimization. *Annals of Operations Research*, 45(1):59–76.

[16] Davis, P. K. (2002). *Analytic architecture for capabilities-based planning, mission-system analysis, and transformation*. Rand Corporation.

[17] DCDH 2012 (2012). *Defence Capability Development Handbook 2012*. Department of Defence.

[18] Deb, K. and Jain, H. (2014). An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints. *Evolutionary Computation, IEEE Transactions on*, 18(4):577–601.

[19] Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: Nsga-ii. *Evolutionary Computation, IEEE Transactions on*, 6(2):182–197.

[20] Elsayed, S. M., Sarker, R. A., and Essam, D. L. (2013). Adaptive configuration of evolutionary algorithms for constrained optimization. *Applied Mathematics and Computation*, 222:680–711.

[21] Elsayed, S. M., Sarker, R. A., and Essam, D. L. (2014). A self-adaptive combined strategies algorithm for constrained optimization using differential evolution. *Applied Mathematics and Computation*, 241:267–282.

[22] Fonseca, C. M., Fleming, P. J., et al. (1993). Genetic algorithms for multiobjective optimization: Formulationdiscussion and generalization. In *ICGA*, volume 93, pages 416–423.

[23] Gaidow, S. and Boey, S. (2005). Australian defence risk management framework: A comparative study. Technical report, DTIC Document.

[24] George, E. D. and Farid, S. S. (2008). Strategic biopharmaceutical portfolio development: An analysis of constraint-induced implications. *Biotechnology progress*, 24(3):698–713.

[25] Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Publishing Corporation, Inc, Reading, MA.

[26] Gutjahr, W. J., Katzensteiner, S., Reiter, P., Stummer, C., and Denk, M. (2010). Multi-objective decision analysis for competence-oriented project portfolio selection. *European Journal of Operational Research*, 205(3):670–679.

[27] Horn, J., Nafpliotis, N., and Goldberg, D. E. (1994). A niched pareto genetic algorithm for multiobjective optimization. In *Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the First IEEE Conference on*, pages 82–87. Ieee.

[28] Hurley, W. and Schobel, K. (2009). Selecting low expenditure defence projects: an estimate of the value of optimization relative to ad hoc procedures. *Military Operations Research*, 14(4):41–46.

[29] Iamratanakul, S., Patanakul, P., and Milosevic, D. (2008). Project portfolio selection: From past to present. In *Management of Innovation and Technology, 2008. ICMIT 2008. 4th IEEE International Conference on*, pages 287–292.

[30] Ishibuchi, H., Sakane, Y., Tsukamoto, N., and Nojima, Y. (2009). Evolutionary many-objective optimization by nsga-ii and moea/d with large populations. In *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on*, pages 1758–1763.

[31] Kangaspunta, J., Liesiö, J., and Salo, A. (2012). Cost-efficiency analysis of weapon system portfolios. *European Journal of Operational Research*, 223(1):264–275.

[32] Khalili-Damghani, K., Tavana, M., and Sadi-Nezhad, S. (2012). An integrated multi-objective framework for solving multi-period project selection problems. *Applied Mathematics and Computation*, 219(6):3122–3138.

[33] Knowles, J. D. and Corne, D. W. (2000). Approximating the nondominated front using the pareto archived evolution strategy. *Evolutionary computation*, 8(2):149–172.

[34] LaBrosse, M. (2010). Project-portfolio management. *Employment relations today*, 37(2):75–79.

[35] Liesi, J., Mild, P., and Salo, A. (2008). Robust portfolio modeling with incomplete cost information and project interdependencies. *European Journal of Operational Research*, 190(3):679 – 695.

[36] Liesiö, J. and Salo, A. (2012). Scenario-based portfolio selection of investment projects with incomplete probability and utility information. *European Journal of Operational Research*, 217(1):162–172.

[37] Lockett, A. G. and Freeman, P. (1970). Probabilistic networks and r & d portfolio selection. *Operational Research Quarterly*, pages 353–359.

[38] Markowitz, H. (1952). Portfolio selection*. *The Journal of Finance*, 7(1):77–91.

[39] Morris, P. and Pinto, J. K. (2010). *The Wiley guide to project, program, and portfolio management*, volume 10. Wiley. com.

[40] Oxford University Press (2013). Oxford Dictionaries Online.

[41] Ponsich, A., Jaimes, A. L., and Coello, C. A. C. (2013). A survey on multiobjective evolutionary algorithms for the solution of the portfolio optimization problem and other finance and economics applications. *Evolutionary Computation, IEEE Transactions on*, 17(3):321–344.

[42] Powell, W. B. (2007). *Approximate Dynamic Programming: Solving the curses of dimensionality*, volume 703. John Wiley & Sons.

[43] Rafiee, M. and Kianfar, F. (2011). A scenario tree approach to multi-period project selection problem using real-option valuation method. *The International Journal of Advanced Manufacturing Technology*, 56(1-4):411–420.

[44] Rechenberg, I. (1973). *Evolutions strategie: Optimierung Technischer Systeme nach Prinzipien der biologischen Evolution*. Fromman-Holzboog, StuttgartA.

[45] Saxena, D. K., Zhang, Q., Duro, J. A., and Tiwari, A. (2011). *Evolutionary Multi-Criterion Optimization: 6th International Conference, EMO 2011, Ouro Preto, Brazil, April 5-8, 2011. Proceedings*, chapter Framework for Many-Objective Test Problems with Both Simple and Complicated Pareto-Set Shapes, pages 197–211. Springer Berlin Heidelberg, Berlin, Heidelberg.

[46] Schaffer, J. D. (1985). Multiple objective optimization with vector evaluated genetic algorithms. In *Proceedings of the 1st International Conference on Genetic Algorithms*, pages 93–100, Hillsdale, NJ, USA. L. Erlbaum Associates Inc.

[47] Shafi, K., Bender, A., and Abbass, H. A. (2011). Fleet estimation for defence logistics using a multi-objective learning classifier system. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, pages 1195–1202. ACM.

[48] Shakhsi-Niaei, M., Iranmanesh, S. H., and Torabi, S. (2014). Optimal planning of oil and gas development projects considering long-term production and transmission. *Computers & Chemical Engineering*, 65:67–80.

[49] Storn, R. and Price, K. V. (1995). Differential evolution-a simple and efficient adaptive scheme for global optimization over continuous spacesn. Technical Report TR-95-012, 1995, Company Secretaries of India, Chennai, Tamil Nadu.

[50] Vandepeer, C., Moon, T., and De Visser, G. (2013). Linking missions to scenarios for analysis of military macro-systems. *OR Insight*, 26(1):47–70.

[51] von Lücken, C., Barán, B., and Brizuela, C. (2014). A survey on multi-objective evolutionary algorithms for many-objective problems. *Computational Optimization and Applications*, 58(3):707–756.

[52] Watkins, C. J. and Dayan, P. (1992). Q-learning. *Machine learning*, 8(3-4):279–292.

[53] Yang, X. (2006). Improving portfolio efficiency: A genetic algorithm approach. *Computational Economics*, 28(1):1–14.

[54] Yu, L., Wang, S., Wen, F., and Lai, K. K. (2012). Genetic algorithm-based multi-criteria project portfolio selection. *Annals of Operations Research*, 197(1):71–86.

[55] Zhang, Q. and Li, H. (2007). Moea/d: A multiobjective evolutionary algorithm based on decomposition. *Evolutionary Computation, IEEE Transactions on*, 11(6):712–731.

[56] Zitzler, E., Laumanns, M., Thiele, L., Zitzler, E., Zitzler, E., Thiele, L., and Thiele, L. (2001). Spea2: Improving the strength pareto evolutionary algorithm.

Acknowledgment

This work was supported by a Defence Related Research (DRR) Grant at UNSW Canberra. The authors are thankful for UNSW Canberra and Australian Defence Force Academy (ADFA) to provide this support for this research.

7. Appendicies

7.1. Appendix I - Evolutionary Multi Objective Optimization

Formally, a multi-objective optimization problem is generally represented as follows:

$$minF(x) = (f_1(x), ..., f_m(x))^T$$

Subject to:

$$x \in \Omega \tag{20}$$

where x is a vector of N decision variables, $f_k(x)$ the objective function of the k^{th} criterion, and Ω the decision space.

The best trade-off among the objectives can be defined by Pareto optimality. Let u, $v \in \Re^k$, u is said to dominate v if and only if $u_k = v_k$, where $k \in 1, ..., m$ and $u_j < v_j$ for at least one index $j \in 1, ..., m$. A point $x^* \in \Omega$ is Pareto optimal if there is no other point $x \in \Omega$ such that $F(x)$ dominates $F(x^*)$. $F(x^*)$ is called a Pareto-objective value.

Multi-Objective Evolutionary Algorithms (MOEAs) are population based meta-heuristic methods that mimic the evolutionary process in natural systems and evolve a set of Pareto optimal solutions. The common steps of an Evolutionary Algorithm (EA) include initialisation of a population of solutions, evaluation of individuals, fitness based selection of parent solutions and generation of new solutions using recombination and mutation operators. A variety of EAs have been proposed including the infamous genetic algorithms [25], differential evolution [49] and evolutionary strategy [44].

From the literature, the most well-known MOEA include vector evaluation genetic algorithm [46], Multi-Objective Genetic Algorithm (MOGA) [22], Niched Pareto Genetic Algorithm (NPGA) [27], Non-dominated Sorting Genetic Algorithm (NSGA, NSGAII) [19], Strength Pareto Evolutionary Algorithm (SPEA) [56], Pareto Archived Evolution Strategy (PAES) [33] and Multi-Objective Evolutionary Algorithm based on Decomposition (MOEAD) [55].

The most recent of these algorithms, MOEA/D and its variants, have been successfully applied to several real-world problems and has shown their competence in comparison to other extant approaches.

7.1.1. MOEA/D

The main idea behind MOEA/D is to decompose the multi-objective optimization problem into a number of single objective optimization problems. A scalarizing function, such as Tchebycheff, is used to decompose the problem into multiple scalar optimization problems. Formally,

$$\min g(x|\lambda, z^*) = \max \lambda_i |f_i(x) - z^*|$$

Subject to:

$$x \in \Omega \tag{21}$$

where $\lambda = (\lambda_1, \cdot \cdot \cdot, \lambda_m)$ is a weight vector representing the weights for each of the m objectives to be optimized, such that $\sum_{(i=1)}^{(\lambda)} \lambda_i = 1$ and $z^* = (z_1^*, \cdot, z_m^*)$ is a vector

representing the ideal objective values for m objectives. Given a set of weight vectors $\lambda^1, \cdots, \lambda^n$, MOEA/D minimizes all of these objective functions in a single run.

To utilise the relations between neighbourhood among all sub problems, optimization of a sub-problem is done by using the information from its neighbouring sub-problems. Hence, in MOEA/D, the closest weight vectors in $\lambda^1, \cdots, \lambda^n$ to a weight vector λ^i constitute the neighbourhood of λ^i . The neighbourhood of the sub-problem consist of all the sub-problems with the weight vectors from the neighbourhood of λ^i. The main steps of MOEA/D are as follows:

input:

1. A stopping criterion;
2. sub: the number of the sub problems considered;
3. a uniform spread of sub weight vectors: $\{\lambda^1, \cdots, \lambda^{sub}\}$;
4. CN: the number of the weight vectors in the neighbourhood of each weight vector.

Step 1: Initialization

1. Compute the Euclidean distance between any two weight vectors and then work out the CN closest weight vectors to each weight vector. For each i = 1, . . . , sub, set $B(i) = i_1, ..., i_{CN}$ where $\{\lambda^{i_1}, ...\lambda^{i_{sub}}\}$ are the sub closest weight vectors to λ^i.
2. Generate an initial sub individuals.
3. Initialize $z = (z_1, ..., z_m)$ by setting $z_i = min\{f_i(x^1), ..., f_i(x^{sub})\}$ and/or $min\{vio_i(x^1), ..., vio_i(x^{sub})\}$.

Step 2 Select mating pool.
Step 3 for each individual, generate a new solution using any EA; Evaluate it based on the fitness function and/or constraint violation and update z_i.

Step 4 Stopping Criteria If the stopping criteria is met, then stop and output $\{x^1, ..., x^{sub}\}$ and the objective value and violation.