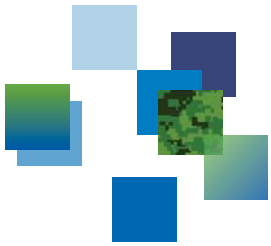
Defence Research and Development Canada
Recherche et développement pour la défense Canada

DRDC | RDDC



On the use of simulated annealing to optimize maintenance schedules for naval platforms

Meghan Green
Jean-Denis Caron
Van Fong
Maritime Operational Research Team (MORT)
DRDC – Centre for Operational Research and Analysis

Defence Research and Development Canada

Reference Document
DRDC-RDDC-2016-D084
December 29, 2016

Canada

On the use of simulated annealing to optimize maintenance schedules for naval platforms

Meghan Green
Jean-Denis Caron
Van Fong
Maritime Operational Research Team (MORT)
DRDC – Centre for Operational Research and Analysis

Defence Research and Development Canada
Reference Document
DRDC-RDDC-2016-D084
December 29, 2016

Abstract

In the summer of 2016, the Maritime Operational Research Team (MORT) hired a student through the Federal Student Work Experience Program (FSWEP) to develop a toolset for generating fleet availability schedules. The blocks of availability include both Standard (or Normal) and High Readiness periods while ships are considered unavailable when in maintenance (long or short term periods) and while in the tiered-readiness program. The toolset, created in R, uses simulated annealing optimization technique. Some of the toolset input parameters include the fleet size and the Operations and Maintenance (O&M) profile of interest for the fleet. This document serves as a summary of the work that was accomplished during the 10-week period. It describes the motivation, defines the problem, and provides some implementation details which are illustrated with a simple example.

Résumé

À l'été 2016, l'Équipe de recherche opérationnelle de la Marine a embauché un étudiant dans le cadre du Programme fédéral d'expérience de travail étudiant pour élaborer des outils visant à générer les calendriers de disponibilité de la flotte. Les blocs de disponibilité comportent des périodes de disponibilité opérationnelle normale (ou standard) et élevée lorsque des navires sont considérés comme étant non disponibles pendant leur maintenance (longues ou courtes périodes) et pendant le programme de préparation opérationnelle échelonnée. Les outils, créés en R, utilisent la technique d'optimisation de type 'Simulated Annealing'. Les paramètres d'entrée des outils comprennent, entre autre, la taille de la flotte et le profil d'intérêt des opérations et de la maintenance de la flotte. Ce document est un résumé des travaux réalisés au cours d'une période de 10 semaines. Il décrit la motivation, définit le problème et fournit certains détails de mise en œuvre illustrés par un exemple simple.

Table of contents

Abstract i

Résumé i

Table of contents ii

List of figures iv

List of tables vi

1 Introduction 1

1.1 Background 1

1.2 Issue 3

1.3 Objective of the work 5

1.4 Scope of the work 5

1.5 This document 5

2 Overview of the fleet scheduling toolset 7

2.1 What is simulated annealing? 7

2.2 Implementation in R 8

2.2.1 Example used for illustration 8

2.2.2 Operations and Maintenance (O&M) profile 9

2.2.3 Representation of a solution as a vector 9

2.2.4 Supporting functions—Conversions and plotting 10

2.2.5 Main file 11

2.2.6 Parameters for the simulated annealing 12

2.3 Example 14

2.3.1 Fitness function 14

2.3.2 Setting the bounds 15

2.3.3 Example results 16

2.4 Discussion 17

3 Conclusion 19

References 20

Annex A: Description of the R files 21

A.1 File main.R 21

A.2 File iterative.R 24

A.3 File importparameters.R 25

A.4 File createMP.R 26

A.5 File forallopen.R 28

A.6 File preplot.R 30

A.7 File diagmat.R 31

A.8 File multiplot.R 32

A.9 File plotfleet.R 33

A.10 File conversions.R 34

A.11 File idealbound.R 35

A.12 File fitness.R 37

A.13 Additional notes 42

Annex B: Input parameters 43

B.1 Worksheet `parameters`—The main parameters 43

B.2 Worksheet `rgb`—Possible platform status and associated RGB colors . 44

B.3 Worksheet `manualprofile` 45

B.4 Worksheet `bound`—Setting individual platform status lower and upper bound 45

B.5 Worksheet `plotorder` 46

List of acronyms 47

List of figures

Figure 1: Typical O&M profile for the Royal Canadian Navy (RCN) Halifax-class frigate. 1

Figure 2: 10-year fleet plan, 13 platforms with Halifax-class frigate O&M profile. . . . 2

Figure 3: Number of platforms in each status—10-year fleet plan, 13 platforms with Halifax-class frigate O&M profile. 2

Figure 4: Example of a 10-year fleet plan (13 platforms with Halifax-class frigate O&M profile) in which, the number of platforms at High Readiness (HR) is not constant–varying between 0 and 4. 3

Figure 5: Second example of an Operations and Maintenance (O&M). 4

Figure 6: 10-year fleet plan for the second example of O&M profile. 4

Figure 7: Number of platforms in each status—10-year fleet plan, 13 platforms with second O&M profile. 5

Figure 8: Pseudo-code for simulated annealing heuristic. 8

Figure 9: Input file for the O&M profile. 9

Figure 10: 10-year fleet plan for $S = (0, 12, 24, 60, 60, 60, 60, 60, 60, 60, 60, 60, 60)$. . . . 10

Figure 11: Number of platforms in each status for solution vector S. 11

Figure 12: Conversion of S to a matrix. 12

Figure 13: Fitness function example. 14

Figure 14: Example of `somme8` vector. 15

Figure 15: Bounds used in the example. 16

Figure 16: Example—10-year fleet plan found by the simulated annealing. 16

Figure 17: Example—Number of platforms in each status. 17

Figure 18: Example of an HR block shifted to the left. 17

Figure B.1: Input worksheet—`parameters` (Part I). 43

Figure B.1: Input worksheet—`parameters` (Part II) 44

Figure B.2: Input worksheet—`rgb`. 44

Figure B.3: Input worksheet—`manualprofile`. 45

Figure B.4: Input worksheet—`bound`. 45

Figure B.5: Input worksheet —`plotorder`. 46

List of tables

Table 1: Input parameters for the *GenSA* call. 13

1 Introduction

1.1 Background

The Maritime Operational Research Team (MORT)[1] has conducted many fleet mix and fleet structure studies for the Royal Canadian Navy (RCN) in the past. Several examples can be found in References [1–5]. One important factor considered in all the studies is the Operations and Maintenance (O&M) profile in place for the fleet. Figure 1 shows the typical O&M profile for the RCN Halifax-class frigate, as outlined in the "Readiness and Sustainment Policy" [6].

During its five-year Operational Cycle (OPCYCLE), each platform goes through a Long-Term Maintenance (LTM) period where the platform is at Extended Readiness (ER). Following the LTM, it falls into Restricted Readiness (RR) status, which applies predominantly for a platform within the Tiered Readiness Programme (TRP).[2] Once TRP is completed, the platform becomes available and is either at:

1. Standard Readiness (SR) also called Normal Readiness (NR) where it can be employed for the purposes of conducting core naval training and executing assigned continental and expeditionary missions that do not entail the possibility of high intensity; or
2. High Readiness (HR), capable of conducting the full-spectrum of combat operations and of being deployed as part of a Task Group (TG).

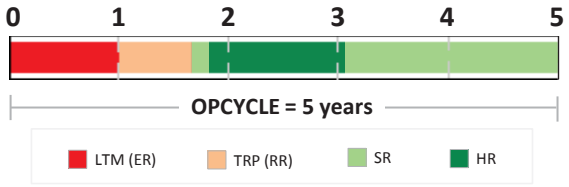


Figure 1: *Typical O&M profile for the RCN Halifax-class frigate.*

For most fleet mix studies, a 10-year fleet plan providing a steady number of platforms available at any given time is used as a starting point. When using an O&M profile such as the one shown in Figure 1, which is comprised of two distinct blocks, i.e. one block of unavailability of 21 months (ER and RR in Figure 1) followed by one block of availability of 39 months (SR and HR in Figure 1), the 10-year fleet plan can be generated by evenly staggering the start of each platform's O&M cycle equally within the fleet. This is done by spacing the O&M cycle by a constant amount of time equal to the number of days in the OPCYCLE divided by the number of platforms. An example for a fleet of 13 platforms with each platform spending 12 months at HR is presented Figure 2.

[1] MORT is comprised of Defence Research and Development Canada (DRDC) Centre for Operational Research and Analysis (CORA) scientists providing operational research and analysis support to Director General Naval Force Development (DGNFD). The main author of this paper was a student, hired through the FSWEP, who worked for MORT in the summer of 2016.

[2] TRP aims to generate ships from ER to SR.

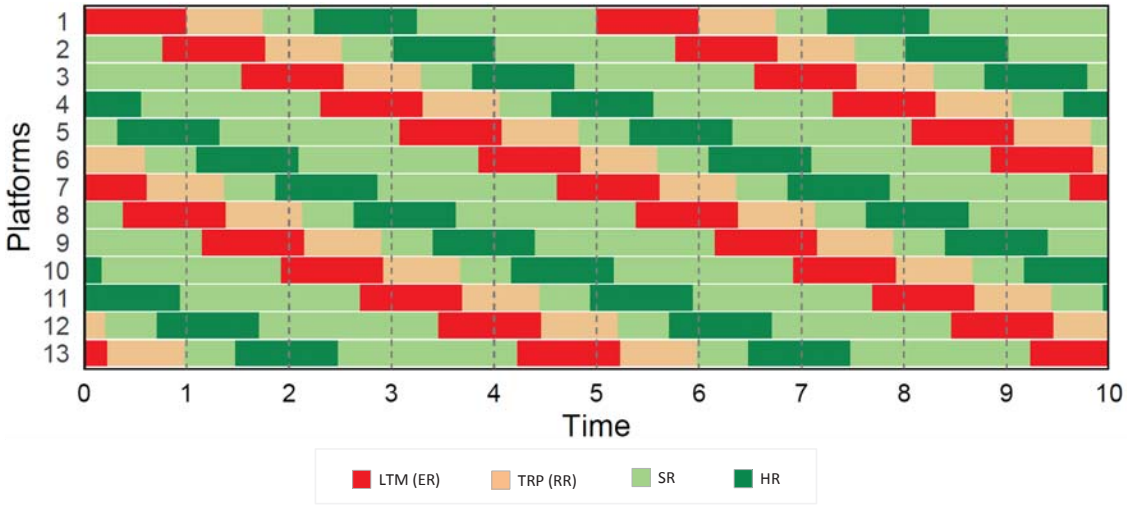


Figure 2: *10-year fleet plan, 13 platforms with Halifax-class frigate O&M profile.*

The 10-year fleet plan shown in Figure 2 allows for a steady maintenance workflow and keeps the number of platforms available as constant as possible. This is illustrated in Figure 3, which shows that at any given time, there are always two or three platforms at HR, and four or five platforms unavailable in maintenance (ER and RR).

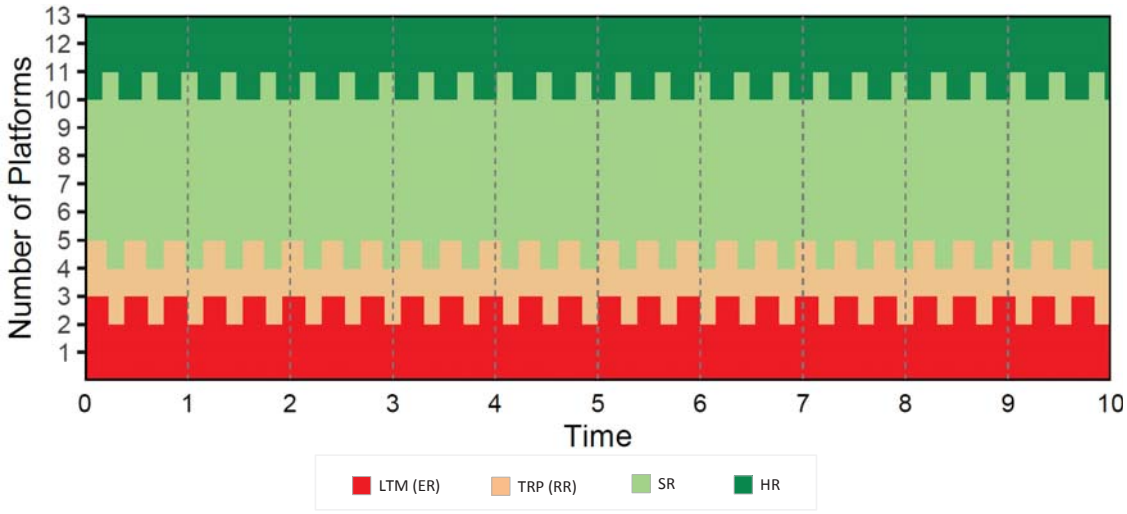


Figure 3: *Number of platforms in each status—10-year fleet plan, 13 platforms with Halifax-class frigate O&M profile.*

In this paper, *constant* refers to achieving the minimum range (i.e. variability) of platform numbers over time thereby being as close to the daily average as possible. Let $\bar{X}$ be the average number of platforms at HR at any given time, which can be calculated a priori based on the number of platforms in the fleet and how long each platform spends at HR in its OPCYCLE.

For the example in Figure 2, i.e. a fleet of 13 ships with each platform at HR for 12 months during its OPCYCLE (i.e. 20% of the time), there is an average $\bar{X} = 2.6$ ships at HR. This can be calculated simply as follows:

$$\bar{X} = 13 \text{ ships} \times 0.2 = 2.6 \text{ ships}$$

In this case, because the number of platforms available on a single day can only be an integer value, *constant* means that the number of platforms is either 2 or 3. Figure 4 shows another example of a schedule with 13 ships spending 12 months at HR for which the number of platforms at HR at any given time is not *constant* (as per the explanation above). In this example, the average number of ships at HR on any given day is also 2.6, but the daily number of ships varies between 0, 1, 2, 3 and 4. For example, no ships are at HR at the end of Year 2 while 4 ships are at HR at the end of Year 10.

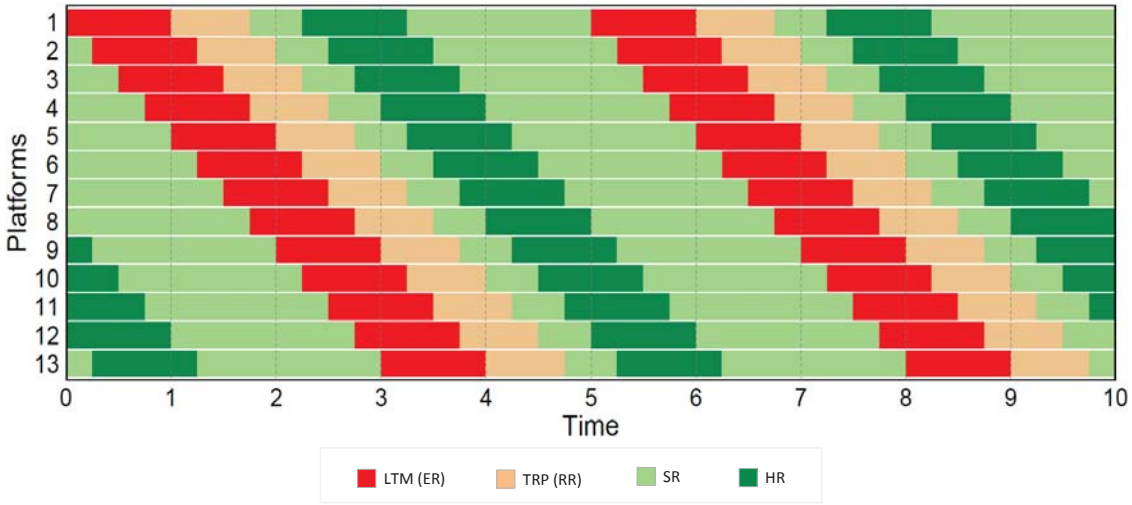


Figure 4: *Example of a 10-year fleet plan (13 platforms with Halifax-class frigate O&M profile) in which, the number of platforms at HR is not constant–varying between 0 and 4.*

A detailed model such as Tyche, which was used in Fleet Mix Studies I and II [1, 2], can then take a 10-year fleet plan (providing a constant flow of ships) as the available platform supply and try to schedule the tasks[3] as they arise in a Monte Carlo fashion.

1.2 Issue

The RCN is exploring new O&M profiles for future classes of ships that are more complex in nature than the one in Figure 1. An example is provided in Figure 5. In this 10-year OPCYCLE, a platform is available 80% of the time in four distinct periods of two consecutive years, represented in green at the beginning of Year 1. During the "Home Game" (lighter green), the platforms are reserved for domestic operations, including serving as Ready Duty Ship (RDS), exercises, training and potentially surge capacity. During the "Away Game" (darker green), the platforms

[3] In Tyche, the demand is formed from a set of maritime vignettes that cover a range of Force Employment, Force Generation, and routine activities expected of the RCN [2].

are forward deployed (or ready to be) and capable of conducting the full spectrum of combat operations. The readiness cycle is binary. A platform is either available when in its “Home” and “Away” periods (green), or unavailable when in maintenance—Deep Maintenance Period (DMP) (red) or Shallow Maintenance Period (SMP) (purple).

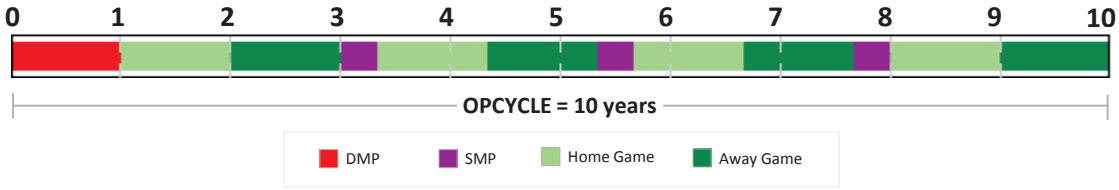


Figure 5: *Second example of an O&M profile.*

Building a 10-year fleet plan with a profile such as the one in Figure 5 by equally spacing the O&M cycle within the fleet may sometimes lead to impractical and inefficient schedules (i.e. not with a constant number of ships). An example of a 10-year plan obtained by equally spacing the O&M cycle on a fleet of 13 platforms is provided is Figure 6.

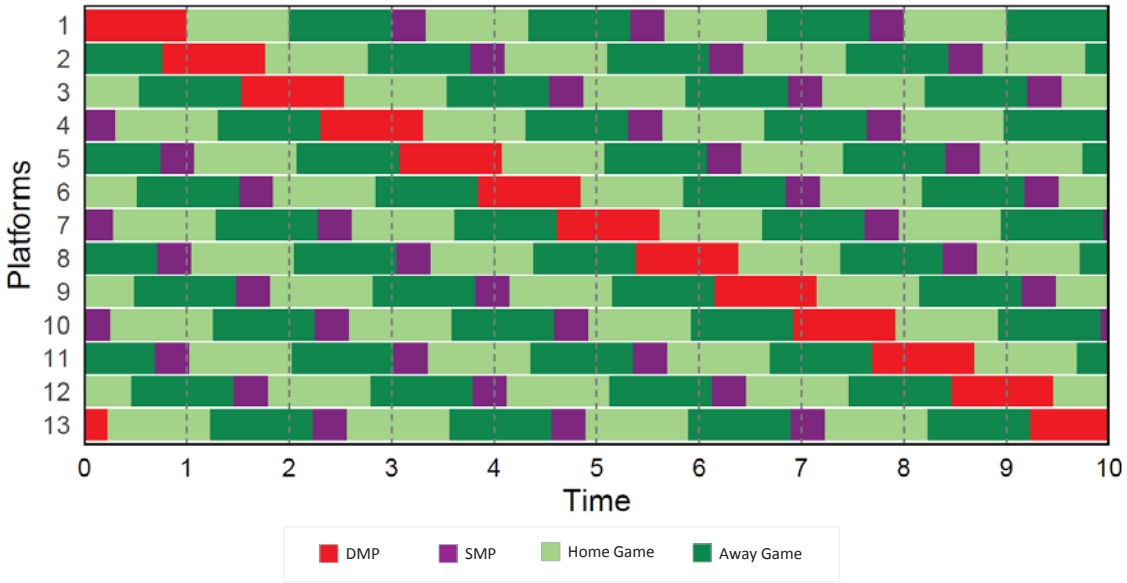


Figure 6: *10-year fleet plan for the second example of O&M profile.*

As mentioned earlier, the goal behind the creation of the 10-year fleet plan is to have a number of platforms available at any given time as constant as possible. As depicted in Figure 7, this is not the case. For instance, at the beginning of Year 1, five platforms are in maintenance, i.e. unavailable, leaving eight platforms available. On the other hand, six months later, 12 out of the 13 platforms are available, with only Platform 1 in DMP.

1.3 Objective of the work

The objective of the work presented in this document was to create a toolset to optimize the O&M schedule in order to build realistic fleet availability schedules. Given various parameters, including the fleet size, the maintenance profile and the OPCYCLE, O&M schedules are created by levelling the number of platforms available at any given time while providing a steady maintenance workflow The toolset includes several supporting functions to visualize O&M schedules (any schedules, initial and resulting), read input from a Microsoft Excel file and conduct analysis of the resulting schedule. The toolset is flexible so that the user can define and customize the objective function of the optimization problem to meet specific needs.

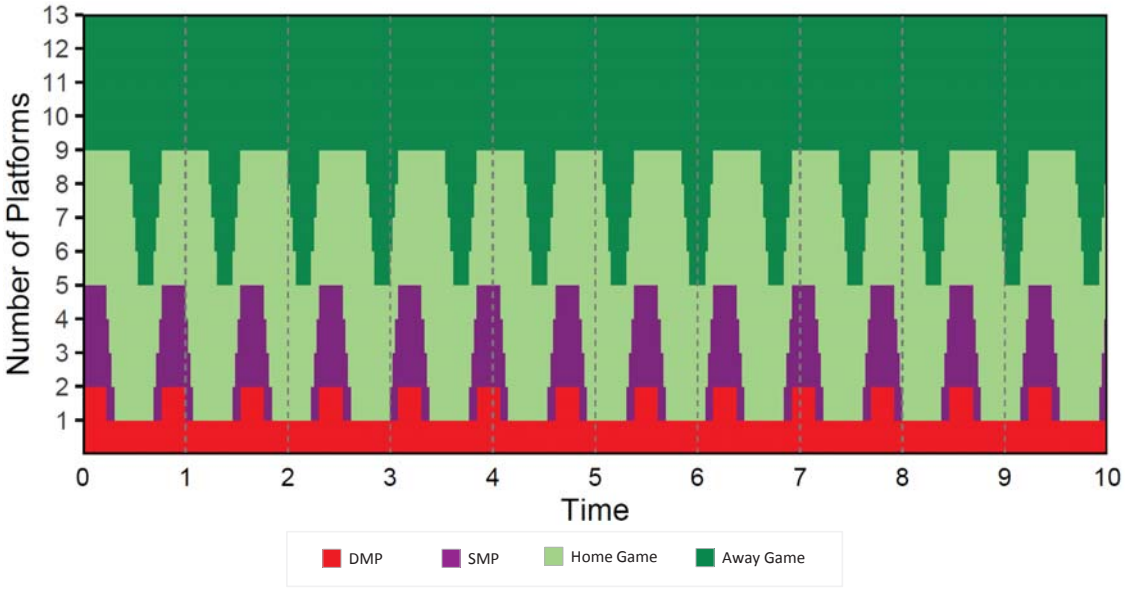


Figure 7: *Number of platforms in each status—10-year fleet plan, 13 platforms with second O&M profile.*

1.4 Scope of the work

There are many different types of optimization techniques that can be used to address the problem described above, including but not limited to: linear integer programming, genetic algorithm, simulated annealing and other optimization heuristics. Only 10 weeks were available to conduct the work, so it was decided that the focus would be on implementing simulated annealing and determine how well it is suited to address the problem.

1.5 This document

This document falls into the DRDC Reference Document (RD) category. An RD is reference material used to support Science and Technology projects and programs, and contains preliminary, limited or no scientific information. The purpose of this RD is to summarize the work

accomplished by a student (i.e. the primary author) hired through the FSWEP who worked in MORT from July to September 2016. The content of the document is informal and is not self-explanatory as many details were omitted. The target audience for this document is analysts who face similar problems and who wish to use or to build upon this work.

What follows in this RD are two subsequent sections and two annexes. Section 2 gives an overview of the toolset and provides some implementation details of the tools which are illustrated using a simple example. Conclusions and proposed areas for future work are summarized in Section 3. Annex A contains a description of the main file and the supporting functions that were developed in R. Finally, Annex B lists the input parameters that are required to run the toolset.

2 Overview of the fleet scheduling toolset

We use simulated annealing to address the optimization problem briefly introduced in Section 1. That is: "Given a fleet size of ships following a particular O&M profile, build a fleet schedule that leads to a steady maintenance workflow and keeps the number of platforms available as constant as possible". The toolset was developed in R, a language and environment for statistical computing and graphics [7]. To evoke the simulated annealing algorithm, the R package called "GenSA" [8] was used. This section provides a description of simulated annealing. An overview of the toolset and how simulated annealing was implemented is also presented and illustrated with a simple example.

2.1 What is simulated annealing?

As explained in [9], simulated annealing is a method for finding a good (not necessarily perfect) solution to an optimization problem that has many solutions or near-solutions. The main principle behind simulated annealing originated from a paper published in 1953 by Metropolis et al [10], in which the algorithm simulated the cooling of material in a heat bath. Thirty years later, the idea presented in [10] was taken by Kirkpatrick et al [11] and applied to optimization problems. The follow-on idea was to use simulated annealing to search for feasible solutions and converge to an optimal solution.

The pseudo-code in Figure 8 presents a common simulated annealing heuristic [12]. It starts from an initial solution s_0, which can be randomly generated or specified by the user. The main while loop stops when the maximum number of iterations is reached or when the fitness value of the solution is deemed acceptable (i.e. a *good enough* solution has been found). The algorithm is driven by three main functions:

1. *Temperature*: The initial temperature is high and slowly decreases as the algorithm runs. With higher temperatures, solutions that are worse than the current solution are kept more often—accepting worse solutions is a fundamental property of meta-heuristics such as simulated annealing because it allows for a more extensive search for the optimal solution.

2. *Neighbour*: The $\mathrm{Neighbour}(s)$ function randomly generates a neighbour solution by making a small change to s, the current solution.

3. *Fitness*: The fitness function, denoted by F in Figure 8, calculates the quality of a solution.

```
s = s_0                                          Initial solution state
fitValue = F(s)                                  Fitness value of initial solution
sBest = s                                        At the start, the best solution is s
fitValueBest = fitValue                          Best fitness value so far is the one for s
k = 0                                            Initialize iteration loop counter
Set kMax                                         Maximum number of iterations
while k < kMax and fitValueBest > AFV            AFV = acceptable fitness value
        T = Temperature(k/kMax)                  Temperature calculation
        sNew = Neighbour(s)                      Pick some neighbour solution (random)
        fitValueNew = F(sNew)                    Compute its fitness value
        if P(s, sNew, T) > Random() then         Should we move to sNew?
                s = sNew                         Yes, change solution state
                fitValue = fitValueNew           Copy over the new fitness value
        end if
        if fitValueNew < fitValueBest then       Is this the best so far?
                sBest = sNew                     Save new neighbour to best found
                fitValueBest = fitValueNew       and the best fitness value
        end if
        k++                                      Increment the iteration loop counter
end while
return sBest                                     Return the best solution found
```

Figure 8: *Pseudo-code for simulated annealing heuristic.*

2.2 Implementation in R

As indicated above, the R package named "GenSA" [8] was used for the simulated annealing optimization. In order to use the package, a main R file (called *main.R*) was created along with many supporting functions required to perform the optimization. Annex A contains a description of the main file and the supporting functions that were developed in R. The description includes: the input parameters, the implicit variables (i.e. global variables), the output and a brief explanation of the steps performed in each of the functions. Note that the R code that was written is not included in this document but can be made available from the authors. Annex B includes a list of all the input parameters that need to be specified by the user via a Microsoft Excel Spreadsheet. The remainder of the section presents some implementation details, referring occasionally to Annexes A and B.

2.2.1 Example used for illustration

In this section, we use the second example presented in Section 1 to illustrate the implementation details. The O&M profile is as shown in Figure 5, i.e. a 10-year OPCYCLE, or 3,600 days.[4] Note that in this section, the words "Away" and "HR" are used interchangeably. A fleet of `nShips` $= 13$ platforms is considered. We also use a timestep of 30 days (1 month), kept in a variable called `Step` (see Figure B.1 in Annex B), which means that the number of periods (called `nPeriods`) in the entire OPCYCLE is 120 (3600 divided by 30).

[4] To simplify the implementation, we assume a year has 360 days and is composed of 12 months of 30 days each.

2.2.2 Operations and Maintenance (O&M) profile

The O&M profile, logged in a vector `profile`, can be specified manually by the user through a worksheet called `manualprofile`. Given a `Step` of 30 days, the input file corresponding to the O&M profile in Figure 5 is shown in Figure 9. The column `type` contains the actual platform status (e.g. DWP, RR, NR, etc.). The possible values are customizable and are user-specified in the Worksheet `rgb`, see Figure B.2 in Annex B. The values under `amount` represent the amount of time (expressed in terms of `Step`) that each platform spends in the corresponding status. For example, the platform spends 12 months in DMP (red), then 12 months at SR (lighter green), followed by 12 months at HR (darker green), etc.

type	amount
2	12
7	12
8	12
9	4
7	12
8	12
9	4
7	12
8	12
9	4
7	12
8	12
	120

Figure 9: *Input file for the O&M profile.*

Inside the R code, the corresponding `profile` vector of length `nPeriods` is as follows:

```
> profile
 [1]  2 2 2 2 2 2 2 2 2 2 2 2 7 7 7 7 7 7 7 7 7 7 7 7 8 8 8 8 8 8
[31] 8 8 8 8 8 8 9 9 9 9 7 7 7 7 7 7 7 7 7 7 7 7 8 8 8 8 8 8 8 8
[61] 8 8 8 8 9 9 9 9 7 7 7 7 7 7 7 7 7 7 7 7 8 8 8 8 8 8 8 8 8 8
[91] 8 8 9 9 9 9 7 7 7 7 7 7 7 7 7 7 7 7 8 8 8 8 8 8 8 8 8 8 8 8
```

2.2.3 Representation of a solution as a vector

The simplest case that the toolset can solve consists of finding when each platform should start its O&M cycle so that the fitness (i.e. quality) of the solution is *good enough*. We will discuss later what the fitness function may look like, but we first start with how a solution is represented. In the example, we are considering a fleet of 13 platforms, which means the solution can be represented as a vector $S = (s_1, s_2, \ldots, s_{13})$, where s_k corresponds the time when the O&M cycle of Platform k starts for all $k \in \{1, 2, \ldots, 13\}$.

In the R code, the solution vector is stored in a variable S. For instance, the variable in R of a solution vector $S = (0, 12, 24, 60, 60, 60, 60, 60, 60, 60, 60, 60, 60)$ is given by

```
> S
 [1]  0 12 24 60 60 60 60 60 60 60 60 60 60
```

In this case, Platforms 1, 2, 3 would start their O&M cycles on timesteps 0, 12 and 24, respectively, while Platforms 4 to 13 would all start their respective cycles in the middle of the OPCYCLE, at timestep 60 (i.e. at the end of Year 5).

2.2.4 Supporting functions—Conversions and plotting

Many supporting functions had to be created. They are described in Annex A. Below is a summary of the most important ones.

Plotting the schedule for the variable S

To visualize the schedule of a solution S with an O&M cycle contained in `profile`, the following two lines of code can be used:

```
> dmi = diagmat(preplot(S,pr=profile))
> plotfleet(dmi)
```

Two graphs are then generated. Using $S = (0, 12, 24, 60, 60, 60, 60, 60, 60, 60, 60, 60, 60)$, the first graph, shown in Figure 10, provides a visual of the 10-year fleet schedule. We can see here, for example, the 10 platforms (4 to 13) that start their O&M cycle at the end of Year 5.

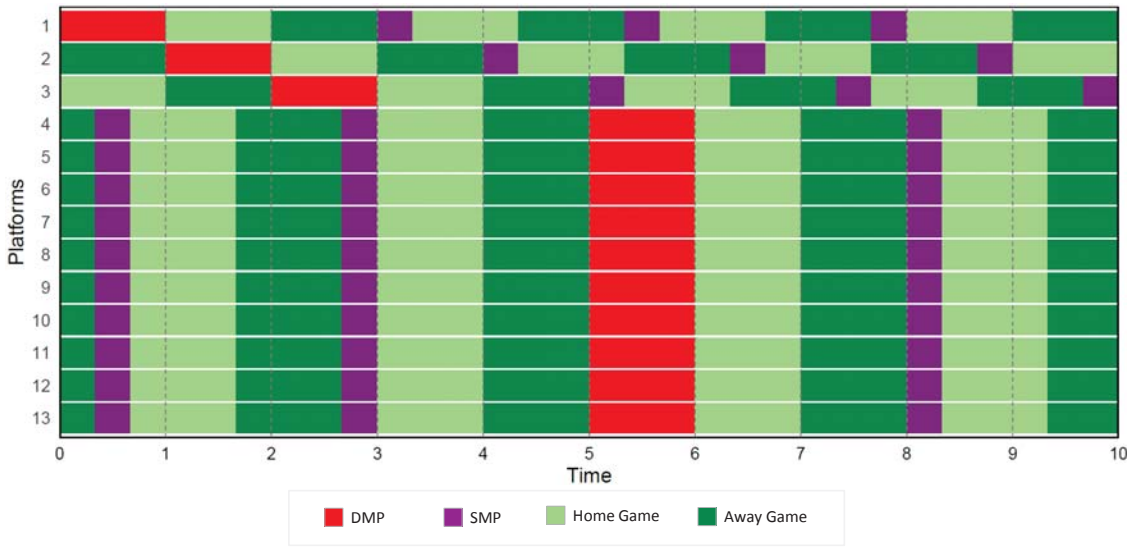


Figure 10: *10-year fleet plan for* $S = (0, 12, 24, 60, 60, 60, 60, 60, 60, 60, 60, 60, 60)$.

The second graph that is created (see Figure 11), similar to the ones presented in Figures 3 and 7, depicts how many platforms are in each status as a function of time. If the objective were to have a steady maintenance workflow, this would not be a very fit solution with 10 platforms in maintenance (red) at the same time for the entire Year 6. The status (i.e. colours) that are displayed in this diagnostic plot, and the order in which they appear, can be specified by the user in the Microsoft Excel input file (see Worksheet `plotorder` in Figure B.5). In addition

to plotting the schedule and diagnostic tool in the R Console, there is an option to save both graphs to a PDF file (see variable `pdf?` in Figure B.1).

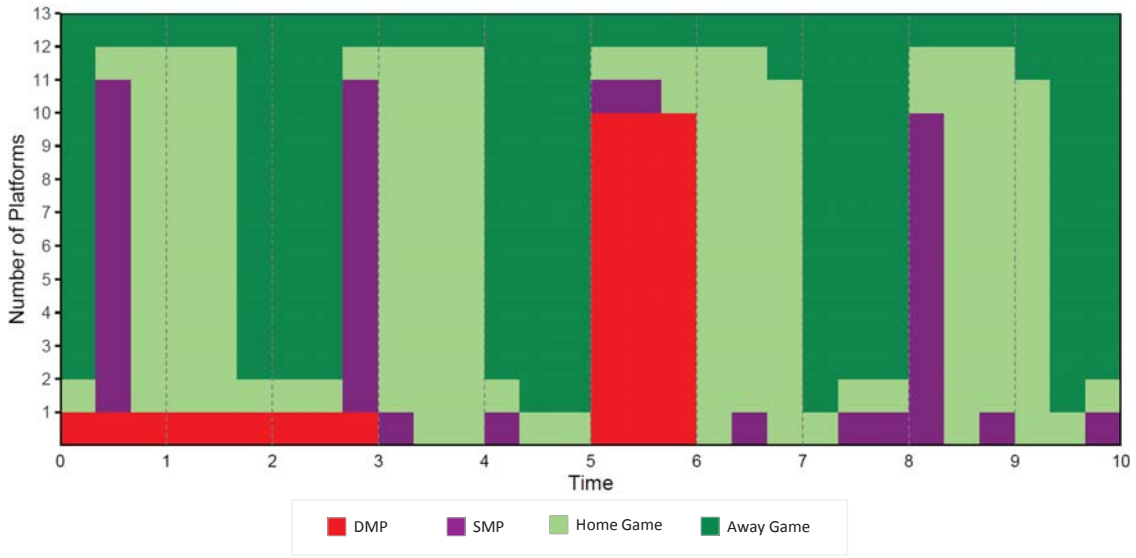


Figure 11: *Number of platforms in each status for solution vector S.*

Conversion of S in a matrix form

The vector S, which contains only the start times for the *k* ships, can be converted to a matrix representing the entire fleet schedule by using the line of R code at the top of Figure 12. The function *preplot* with parameters `S` and `pr` converts the vector to a dataframe which is then converted to a matrix using the *d2m* function. As shown in Figure 12, the result is a matrix (`sch` in this example) with `nShips` rows and `nPeriods` columns. Such a matrix may be used in the fitness function to evaluate the quality of a solution (e.g. to compute how often there are more than n platforms in maintenance).

2.2.5 Main file

The main file that drives the optimization process can be broken down in the following 11 sub-processes:

1. **Clear workspace and import packages**: Clears the workspace (all variables and functions) and imports all the packages required to run the main code and the supporting functions.
2. **Import necessary functions**: Loads the user-defined functions.
3. **Import parameters**: Reads all input parameters, i.e. the five worksheets (`parameters`, `bound`, `manualprofile`, `rgb` and `plotorder`) as presented in Annex B.
4. **Initial solution and profile setup**: Set up the initial solution `S` and `profile` variables based on input parameters.

5. **Clean and print**: Assigns values to and resizes internal variables. It also displays input parameters and variables in the R Console for user validation.

6. **Initial plot**: Plots initial solution and diagnostic graph in the R Console and to a file (if desired).

7. **Boundary variables**: Sets the boundary values for the acceptable range of number of platforms in each state. This can be used in the fitness function calculations. This is discussed later in this section.

8. **Run the simulated annealing**: Runs the simulated annealing. Details are provided below.

9. **Solution clean up**: Post-processes the results of the simulated annealing.

10. **Improvement**: Shows performance values including the first fitness value, the final fitness value, and the ratio between them, where a higher value means improved fitness.

11. **Final plot**: Plots the best solution found by the simulated annealing.

```
> sch = d2m(preplot(S,pr = profile))
> sch
   1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45
1  2 2 2 2 2 2 2 2 2  2  2  2  7  7  7  7  7  7  7  7  7  7  7  8  8  8  8  8  8  8  8  8  8  8  8  9  9  9  9  7  7  7  7  7
2  8 8 8 8 8 8 8 8 8  8  8  8  2  2  2  2  2  2  2  2  2  2  2  7  7  7  7  7  7  7  7  7  7  7  8  8  8  8  8  8  8  8  8
3  7 7 7 7 7 7 7 7 7  7  7  7  8  8  8  8  8  8  8  8  8  8  8  8  2  2  2  2  2  2  2  2  2  2  2  7  7  7  7  7  7  7  7
4  8 8 8 8 9 9 9 9 7  7  7  7  7  7  7  7  7  7  7  8  8  8  8  8  8  8  8  8  8  8  8  8  9  9  9  9  7  7  7  7  7  7  7  7
5  8 8 8 8 9 9 9 9 7  7  7  7  7  7  7  7  7  7  7  8  8  8  8  8  8  8  8  8  8  8  8  8  9  9  9  9  7  7  7  7  7  7  7  7
6  8 8 8 8 9 9 9 9 7  7  7  7  7  7  7  7  7  7  7  8  8  8  8  8  8  8  8  8  8  8  8  8  9  9  9  9  7  7  7  7  7  7  7  7
7  8 8 8 8 9 9 9 9 7  7  7  7  7  7  7  7  7  7  7  8  8  8  8  8  8  8  8  8  8  8  8  8  9  9  9  9  7  7  7  7  7  7  7  7
8  8 8 8 8 9 9 9 9 7  7  7  7  7  7  7  7  7  7  7  8  8  8  8  8  8  8  8  8  8  8  8  8  9  9  9  9  7  7  7  7  7  7  7  7
9  8 8 8 8 9 9 9 9 7  7  7  7  7  7  7  7  7  7  7  8  8  8  8  8  8  8  8  8  8  8  8  8  9  9  9  9  7  7  7  7  7  7  7  7
10 8 8 8 8 9 9 9 9 7  7  7  7  7  7  7  7  7  7  7  8  8  8  8  8  8  8  8  8  8  8  8  8  9  9  9  9  7  7  7  7  7  7  7  7
11 8 8 8 8 9 9 9 9 7  7  7  7  7  7  7  7  7  7  7  8  8  8  8  8  8  8  8  8  8  8  8  8  9  9  9  9  7  7  7  7  7  7  7  7
12 8 8 8 8 9 9 9 9 7  7  7  7  7  7  7  7  7  7  7  8  8  8  8  8  8  8  8  8  8  8  8  8  9  9  9  9  7  7  7  7  7  7  7  7
13 8 8 8 8 9 9 9 9 7  7  7  7  7  7  7  7  7  7  7  8  8  8  8  8  8  8  8  8  8  8  8  8  9  9  9  9  7  7  7  7  7  7  7  7
   46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87
1   7  7  7  7  7  7  7  8  8  8  8  8  8  8  8  8  8  8  8  9  9  9  9  7  7  7  7  7  7  7  7  7  7  7  8  8  8  8  8  8  8  8
2   8  8  8  9  9  9  9  7  7  7  7  7  7  7  7  7  7  7  7  8  8  8  8  8  8  8  8  8  8  8  8  8  9  9  9  9  7  7  7  7  7  7
3   7  7  7  8  8  8  8  8  8  8  8  8  8  8  8  8  9  9  9  9  7  7  7  7  7  7  7  7  7  7  7  8  8  8  8  8  8  8  8  8  8  8
4   7  7  7  8  8  8  8  8  8  8  8  8  8  8  8  8  2  2  2  2  2  2  2  2  2  2  2  7  7  7  7  7  7  7  7  7  7  8  8  8  8  8
5   7  7  7  8  8  8  8  8  8  8  8  8  8  8  8  8  2  2  2  2  2  2  2  2  2  2  2  7  7  7  7  7  7  7  7  7  7  8  8  8  8  8
6   7  7  7  8  8  8  8  8  8  8  8  8  8  8  8  8  2  2  2  2  2  2  2  2  2  2  2  7  7  7  7  7  7  7  7  7  7  8  8  8  8  8
7   7  7  7  8  8  8  8  8  8  8  8  8  8  8  8  8  2  2  2  2  2  2  2  2  2  2  2  7  7  7  7  7  7  7  7  7  7  8  8  8  8  8
8   7  7  7  8  8  8  8  8  8  8  8  8  8  8  8  8  2  2  2  2  2  2  2  2  2  2  2  7  7  7  7  7  7  7  7  7  7  8  8  8  8  8
9   7  7  7  8  8  8  8  8  8  8  8  8  8  8  8  8  2  2  2  2  2  2  2  2  2  2  2  7  7  7  7  7  7  7  7  7  7  8  8  8  8  8
10  7  7  7  8  8  8  8  8  8  8  8  8  8  8  8  8  2  2  2  2  2  2  2  2  2  2  2  7  7  7  7  7  7  7  7  7  7  8  8  8  8  8
11  7  7  7  8  8  8  8  8  8  8  8  8  8  8  8  8  2  2  2  2  2  2  2  2  2  2  2  7  7  7  7  7  7  7  7  7  7  8  8  8  8  8
12  7  7  7  8  8  8  8  8  8  8  8  8  8  8  8  8  2  2  2  2  2  2  2  2  2  2  2  7  7  7  7  7  7  7  7  7  7  8  8  8  8  8
13  7  7  7  8  8  8  8  8  8  8  8  8  8  8  8  8  2  2  2  2  2  2  2  2  2  2  2  7  7  7  7  7  7  7  7  7  7  8  8  8  8  8
   88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120
1   8  8  8  8  8  9  9  9  9  7  7  7   7   7   7   7   7   7   7   7   7   8   8   8   8   8   8   8   8   8   8   8   8
2   7  7  7  7  7  8  8  8  8  8  8  8   8   8   8   8   8   9   9   9   9   7   7   7   7   7   7   7   7   7   7   7   7
3   8  9  9  9  9  7  7  7  7  7  7  7   7   7   7   7   7   8   8   8   8   8   8   8   8   8   8   8   8   9   9   9   9
4   8  8  8  8  8  8  8  8  8  8  9  9   9   9   7   7   7   7   7   7   7   7   7   7   7   7   8   8   8   8   8   8   8
5   8  8  8  8  8  8  8  8  8  8  9  9   9   9   7   7   7   7   7   7   7   7   7   7   7   7   8   8   8   8   8   8   8
6   8  8  8  8  8  8  8  8  8  8  9  9   9   9   7   7   7   7   7   7   7   7   7   7   7   7   8   8   8   8   8   8   8
7   8  8  8  8  8  8  8  8  8  8  9  9   9   9   7   7   7   7   7   7   7   7   7   7   7   7   8   8   8   8   8   8   8
8   8  8  8  8  8  8  8  8  8  8  9  9   9   9   7   7   7   7   7   7   7   7   7   7   7   7   8   8   8   8   8   8   8
9   8  8  8  8  8  8  8  8  8  8  9  9   9   9   7   7   7   7   7   7   7   7   7   7   7   7   8   8   8   8   8   8   8
10  8  8  8  8  8  8  8  8  8  8  9  9   9   9   7   7   7   7   7   7   7   7   7   7   7   7   8   8   8   8   8   8   8
11  8  8  8  8  8  8  8  8  8  8  9  9   9   9   7   7   7   7   7   7   7   7   7   7   7   7   8   8   8   8   8   8   8
12  8  8  8  8  8  8  8  8  8  8  9  9   9   9   7   7   7   7   7   7   7   7   7   7   7   7   8   8   8   8   8   8   8
13  8  8  8  8  8  8  8  8  8  8  9  9   9   9   7   7   7   7   7   7   7   7   7   7   7   7   8   8   8   8   8   8   8
```

Figure 12: *Conversion of S to a matrix.*

2.2.6 Parameters for the simulated annealing

The function *GenSA* available in the "GenSA" R package takes several input parameters. They are all discussed in [8]. This includes the neighbour generating function, the acceptance probability (function P in Figure 8) and the function that determines the temperature to use depending on the amount of time the algorithm has been running.

Unfortunately, there is no magical recipe to follow in selecting these parameters that will yield acceptable solutions for all types of problem. The function *GenSA* uses default values for the parameters that are not specified in the function call. For the examples that were explored in this work, the default values were used for the three aforementioned parameters and in all cases, it led to acceptable solutions. In the literature, there exists a lot of guidelines to help setting up these parameters but we are not discussing any in this document.

The following lines of code in *main.R* calls the simulated annealing optimization.

```
out <- GenSA(par = as.numeric(S), fn = fitf, lower = lower, upper = upper,
             control=list(smooth = FALSE,max.time=TimeOut*60,threshold.stop=0,verbose=TRUE),
             prf = profile)
```

The input parameters are described in Table 1.

Table 1: *Input parameters for the* GenSA *call.*

Parameter	Description
`par`	Initial solution for the vector `S`.
`fn`	Fitness function to minimize. It is defined by the user (an example is presented below). In this case, it is set to `fitf`.
`lower`	Vector (same length as `S`) with lower bounds for the elements of `S`. Values are read from input file (discussed below).
`upper`	Vector (same length as `S`) with upper bounds for the elements of `S`. Values are read from input file (discussed below).
`smooth`	Whether `fn` is smooth, or differentiable almost everywhere in the region of `par`. It is set to FALSE but can be changed depending on the problem.
`max.time`	Maximum run time in seconds.
`threshold.stop`	Stop the execution when the expected objective function value `threshold.stop` is reached.
`verbose`	Whether messages (i.e. traces) are displayed during the execution.
`prf`	The O&M profile. This is a parameter passed to the `fn` function. If other parameters were required by the fitness function, they would simply be added after `prf`.

2.3 Example

2.3.1 Fitness function

The toolset is flexible and lets the user define the fitness function based on the objective of the optimization problem. In this example, we want to build a fleet schedule that will lead to a steady maintenance workflow and keeps the number of platforms available (at HR) as constant as possible. To do so, we need a function that takes vectors `S` and `profile` as input, and that returns an associated fitness value taking into account both maintenance constraints and a desired number of platforms at HR. The fitness function is used to compute the quality of the current solution for every iteration in the simulated annealing, therefore, it is called very often and should be as efficient as possible (in terms of computation time).

An example of a fitness function is provided in Figure 13. First, `S` is converted to a matrix `sch` based on the profile of interest. The variables `somme2` and `somme9` are vectors containing how many ships are in status 2 (DMP) and status 9 (SMP) on any given time period. The total number of ships in maintenance per time period is calculated and stored in the vector `somTotal`. These vectors can be used to compute the number of timesteps where the number of ships in maintenance (or in any maintenance status) falls within the acceptable bounds. For instance, if only three ships can be maintained simultaneously in the dry dock facilities, then solutions where `somTotal` is greater than three would not be given a good score.

```
# Fitness Function - Example
fitness1 <- function(S, prf) {#implicit: {mins,maxs}

  # Converts S to a matrix (sch)
  sch = d2m(preplot(S,pr = prf))

  # Counts how many ships are in Status 2 (LTM)
  somme2 <- colSums(sch==2)
  # Counts how many ships are in Status 9 (STM)
  somme9 <- colSums(sch==9)
  # Counts how many ships are in maintenance (Status 2 or 9)
  somTotal <- somme2 + somme9
  # Counts how many ships are in Status 8 (HR)
  somme8 <- colSums(sch==8)

  # Computes the fitness
  # Part 1 is related to maintenance (Multiplied by 1000 for higher
  # weight) and Part 2 to ships at HR (multiply only by 1)
  fitnessVal <- 1000*sum((somTotal>maxs$Total) | (somme2>maxs$LTM) |
                         (somme2<mins$LTM) | (somme9>maxs$STM)|
                         (somme9<mins$STM)) +
                1*sum((somme8>maxs$HR) | (somme8<mins$HR))

  return(as.numeric(fitnessVal))
}
```

Figure 13: *Fitness function example.*

The total number of ships at HR at any given timestep is also calculated and stored in the vector `somme8`, i.e. the status 8 corresponding to HR in the mapping. Figure 14 provides and example of the `somme8` vector as calculated with the example shown in Figures 10, 11 and 12.

The fitness value calculation is twofold:

```
> somme8
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30
 11  11  11  11   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1  11  11  11  11  11  11  11  11  11  11

 31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60
 11  11   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1  11  11  11  11  12  12  12  12  12  12  12  12

 61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90
  1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   2   2   2   2  12  12  12  12  11  11

 91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120
 11  11  11  11  11  11   1   1   1   1   1   1   1   1   1   1   1   1   2   2   2   2  12  12  12  12  11  11  11  11
```

Figure 14: *Example of* `somme8` *vector.*

1. **Maintenance constraints**: It is the first *sum* in the function shown in Figure 13, which counts how many timesteps the number of ships in status 2 is outside the Long-Term Maintenance (LTM) range [`mins$LTM`, `maxs$LTM`] or the number of ships in status 9 is outside the Short Term Maintenance (STM) range [`mins$STM`, `maxs$STM`], or the number of times the total number of ships in maintenance is greater than `maxs$Total`. Basically, this sum is counting how often one of the maintenance constraints is violated.

2. **Number of ships at HR**: It corresponds to the second *sum* in the fitness function and counts how many timesteps the number of ships at HR is outside the desired limit of [`mins$HR`, `maxs$HR`].

Note that the first sum in the fitness function is multiplied by 1,000 while the second sum, only by 1. This places an increased importance on the maintenance constraints. In its search for a good solution, the simulated annealing algorithm first tries to find solutions where the maintenance constraints are met because it has a higher importance in the fitness calculation.

The problem described in this section can be characterized as a *Constraint Satisfaction Problem*, which in a nutshell, means that the objective is to find a solution where all the constraints will be satisfied. In this case, the fitness function (as in Figure 13) returns a value of 0 when all the constraints are met. That explains why we can set the `threshold.stop` value to 0 in the *GenSA* call function.

2.3.2 Setting the bounds

The bounds (lower and upper) are kept in the vectors `mins` and `maxs`. As discussed in both annexes, these vectors are read from the Microsoft Excel Input File. They can be either set by the user manually or automatically calculated within the function *idealbound* (triggered on/off using the variables `manualbound` or `manual?`). The bounds used as example in this document, which are manually entered, are presented in the table shown in Figure 15. In this case, we aim for a solution where there will be a maximum of three ships in maintenance (see row labeled `Total`), at least one ship in status 2 but not more than two (see row labeled `LTM`) and at least one ship in status 9 but not more than two (see row labeled `STM`). At the same, we want the number of ships in HR to vary between five and six.

bound	Min	Max
Total	0	3
Open	0	0
LTM	1	2
RR	0	0
4	0	0
5	0	0
6	0	0
SR	0	0
HR	5	6
STM	1	2
10	0	0
11	0	0
12	0	0
13	0	0
14	0	0
15	0	0

Figure 15: *Bounds used in the example.*

2.3.3 Example results

We ran the simulated annealing with the parameters and fitness function presented in this section. The solution found is $S = (0, 4, 12, 16, 24, 36, 48, 56, 68, 80, 88, 100, 112)$ and has a corresponding fitness value of 0, meaning that this is a solution in which all the constraints are satisfied. The solution is illustrated in Figure 16.

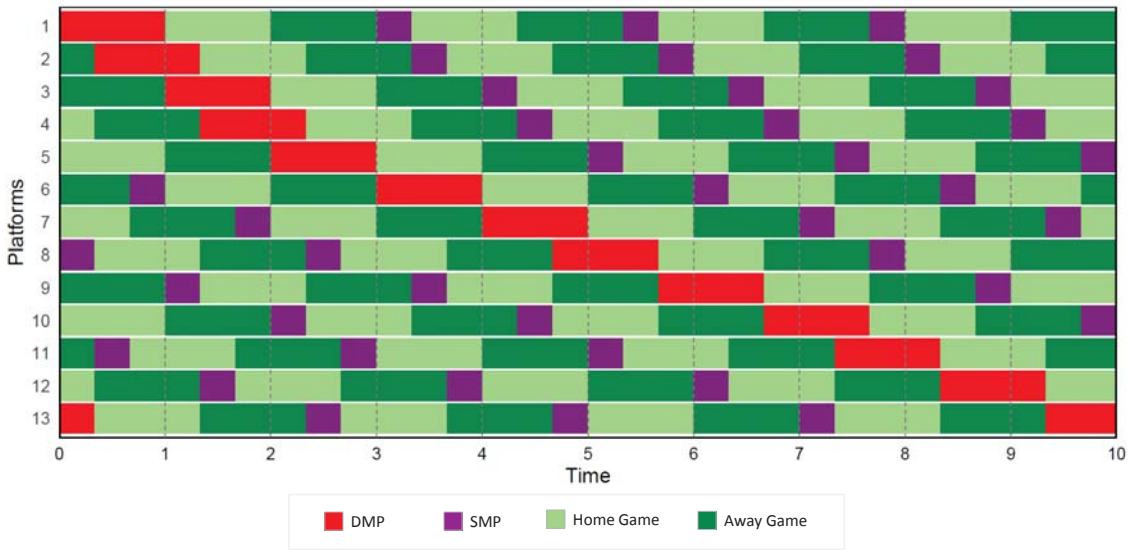


Figure 16: *Example—10-year fleet plan found by the simulated annealing.*

Figure 17 shows how many ships are in each status as a function of time. We can see that all the maintenance constraints are satisfied and that the number of HR ships is nearly constant, varying between between five and six.

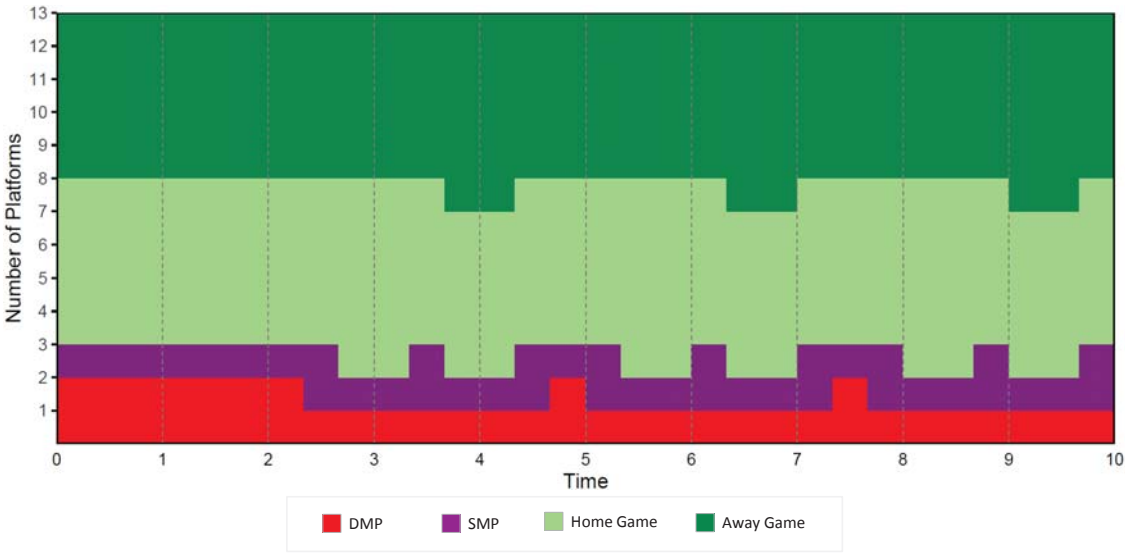


Figure 17: *Example—Number of platforms in each status.*

2.4 Discussion

There are others types of problems that can be solved with the toolset described here. In the example used in this section, the simulated annealing algorithm was only allowed to change the O&M cycle start of each platform, meaning that the actual profile of each platform was fixed (i.e. one year of DMP, followed by one year at SR, then one year at HR, etc.) In reality, there are problems in which the user may want to allow the HR blocks to be shifted to the left within the ship's availability period, as long as it does not fall on a timestep where the ship is scheduled to be in maintenance. This is illustrated in Figure 18, which shows an HR block that has been shifted by six months. The number of time periods the ships are shifted to the left

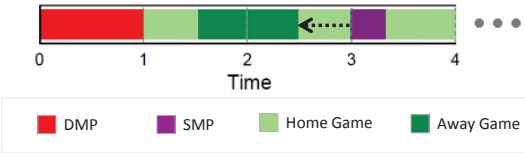


Figure 18: *Example of an HR block shifted to the left.*

may be different from one platform to another. In this case, in addition to finding the start of each platform's O&M cycle, the simulated annealing function would also try to find the start of each HR block within the O&M cycle. In the toolset, the way the vector solution S was designed would be able to accommodate such optimization problems. The length of S would be larger, having to include the number of timesteps each HR block is shifted by (see Annexes A and B for more details). The user would obviously have to create a custom fitness function based on the problem of interest.

It is also possible to address problems like the one presented in [5]. In that case, the objective was to find a 10-year fleet plan where it would be possible to send a task group (i.e. a certain

number of ships transiting together in and out of an area of operation) on any given day in the OPCYCLE while maintaining a particular number of ships at home for RDS. Input parameters were created in the toolset to account for this type of problem (see rows labeled `MinMission` and `nRDSs` in Figure B.1).

Finally, the authors believe that minor modifications (if any) can be made to the toolset to generalize it to address problems not specifically looking at O&M profiles for naval platforms. For instance, the toolset can be used to solve problems involving equipment that would follow an O&M profile similar to the ones presented in this RD. Examples of such equipment may include modules or removable/plug-and-play systems that would have to be tested and maintained (unavailable) and embarked on ships (available) based on an equipment schedule similar to a fleet schedule.

3 Conclusion

The goal of this paper was to document the scheduling toolset developed by the FSWEP student to enable explorations into optimal fleet scheduling. This toolset is flexible and straightforward to modify in R, allowing an open framework for exploring any number of options or concepts related to fleet scheduling, O&M profiles and ship availability. Almost all the parameters in the toolset are easy to adjust, providing a user with maximum flexibility to define different fleet configurations, maintenance profiles, readiness states, cycle lengths, and more. Of particular note, the implementation of simulated annealing and the open nature of the fitness function allows a user to customize the optimization to study a range of questions, from simple optimization of ships within a particular readiness state to more complex scheduling problems involving the optimal scheduling of multiple ships to maximize overall Task Group availability over a 10-year OPCYCLE. Depending on the nature of the question or concept being explored, this R toolset may be a very useful scheduling tool for MORT moving forward.

References

[1] Allen, D., Blakeney, D., Burton, R., and Purcell, L. D., Fleet mix study: determining the capacity and capability of the future naval force structure, Defence R&D Canada, DRDC CORA TR 2005-38, UNCLASSIFIED, December 2005.

[2] Bourque, A., and Eisler, C., Fleet Mix Study Iteration II: Making the Case for the Capacity of the “Navy after Next”, Defence R&D Canada, DRDC CORA TR 2010-159, UNCLASSIFIED, August 2010.

[3] Bourque, A., and Mirshak, R., Effect of platform numbers on the projected capacity of the Canadian Surface Combatant fleet (U), DRDC-RDDC-2015-L040, DRDC CORA, PROTECTED A, February 2015.

[4] Bourque, A., and Mirshak, R., Impact of Canadian Surface Combatant Fleet Size on Delivering Naval Ambitions (U), DRDC-RDDC-2015-R244, DRDC CORA, PROTECTED A, November 2015.

[5] Caron, J.-D., and Fong, V., Evaluating the impact of the Through-life Engineering and Operations (TEO) Concept on CSC operational output (U), DRDC-RDDC-2016-L047, DRDC CORA, PROTECTED A, March 2016.

[6] CFCD 129 Maritime Command Readiness and Sustainment Policy, Department of National Defence, 23 October 2009.

[7] R Core Team (2013), R: A Language and Environment for Statistical Computing, R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0.

[8] Gubian, S., Xiang, Y., Suomela, B., and Hoeng, J. (2016), R Functions for Generalized Simulated Annealing (online), `https://cran.r-project.org/web/packages/GenSA/GenSA.pdf`, CRAN (Accessed: 28 October 2016).

[9] Geltman, K. E. (2014), R Functions for Generalized Simulated Annealing (online), `http://katrinaeg.com/simulated-annealing.html`, CRAN (Accessed: 28 October 2016).

[10] Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, and A.H., Teller (1953), Equation of State Calculation by Fast Computing Machines, Journal of Chemical Physics, 21, 1087-1091.

[11] Kirkpatrick, S , Gelatt, C.D., and Vecchi, M.P. (1983), Optimization by Simulated Annealing. Science, vol 220, No. 4598, 671-680.

[12] Wikipedia, The free encyclopedia (2012), Simulated Annealing (online), `http://users.encs.concordia.ca/~kharma/coen352/Project/SimulatedAnnealing.pdf`, CRAN (Accessed: 31 October 2016).

Annex A Description of the R files

This annex contains a description of the main file and the supporting functions that were developed in R. This documentation was provided by the FSWEP student as part of her final deliverable to MORT and includes the input, the implicit variables (i.e. global variables), the output and a brief explanation of the processes (i.e. steps) performed in each of the functions. Many of the functions presented can be used as supporting functions, e.g. plotting a schedule, reading an O&M profile from a file, and more.

A.1 File main.R

R Script *Last Updated: September 29, 2016*

Section: clear workspace and import packages

- readxl – *importparameters.R*, *iterative.R*
- reshape2 – *importparameters.R*, *plotfleet.R* (internal)
- GenSA – *main.R*
- tictoc – *main.R*
- ggplot2 – *plotfleet.R*
- grid – *multiplot.R* (internal)

Section: import necessary functions

- set working directory to appropriate section of shared drive
- source *createMP.R*
- source *forallopen.R*
- source *preplot.R*
- source *diagmat.R*
- source *multiplot.R*
- source *plotfleet.R*
- source *conversions.R*
- source *idealbound.R*
- source *fitness.R*

Section: import parameters

- source *import-parametersR*
- `fitf` assigns which fitness function is going to be used several times throughout, and cannot be imported with the input function from the spreadsheet.

Section: initialization of S and profile setup

- equalS: creates a starting state vector `S` (starting times of long term maintenance only) that will start the ships equally spaced from one another as long as `starttype` (imported) is equal to one of the elements in `eTypes`.
- zeroS: creates a starting state vector `S` (starting times of long term maintenance only) that starts the ships' maintenance periods all at the same time, as long as `starttype` is one of the elements in `zTypes`.
- randS: creates a starting state vector `S` (starting times of long term maintenance only) that randomly generates long term maintenance start times for the ships, assuming `starttype` is one of the elements in `rTypes`.
- Using `starttype` and `eTypes`, `zTypes`, `rTypes`, assigns the `S` to either `equalS`, `zeroS`, or `randS` (`randS` left aligned and sorted here).
  - This has an error if `starttype` isn't one of the values found.
  - `Unknown Start Type`
- If `manual?` is set to TRUE, the profile is set to `mmp` (imported), if not, it is set to `createMP()` from the values imported.
- If `allopen` is set to TRUE, there will be a high readiness/away period in every open section between two maintenance periods, and `AwayStarts`, `OpenStarts`, `maxshiftrange`, and `nHRs` are reassigned accordingly.
- If `AWAY` is TRUE, the `S` vector is extended to include the start times of each away period, going from a length of `nShips` to `nShips` $\times$ `(1+nHRs)`.
- If `manual?` (again) is TRUE, the `shiftrange` has to be recalculated to be less than the maximum possible shift range either imported or recalculated with `allopen` TRUE.
- If `manualbound` or `manual?` are set to FALSE, then `idealbound()` should be run. Otherwise the boundaries will be left as is from the import.
- One last error check for the manual profiles not being the correct length.
  - `CycleLength does not match profile`

Section: clean and print

- `filename` assigned to include `nShips`, `starttype`, and `both?`.

- If `info?` is TRUE, the output file is created and the concatenate/"cat" printing begins.
- `S` reshaped into matrix form (`m`) with each row a ship and each column an event's start time.
- The row names and column names are assigned to the reshaped matrix for legibility.
- Again if `info?`, then the initial state vector `S` is printed.
  - If `S` is changed to `m` in this line, then it should be printing the matrix version nicely, but it still concatenates it into vector form, just grouped by event instead of by ship. The creation of `m` uses `byrow = TRUE`, and the flattening does not.

Section: initial plot

- PDF name assigned.
- Diagnostic list for initial schedule assigned to `dmi` with initial `S`.
- If `pdf?` is TRUE, then the fleet is plotted to both the file and the console, otherwise plotted to just the console.

Section: boundary variables

- The lower bound for each ship is stored in `pership`, which the value is 0 if there are no high readiness/away values, else is 0 followed by the `AwayStarts` minus an epsilon if `AWAY` is TRUE.
- The lower bound for each `S` parameter set, for each index corresponding to a long term maintenance start time as 0 and each high readiness/away index being the furthest shift left minus an epsilon.
- Upper bound is set as `nPeriods` for the long term maintenance indices and `AwayStarts` plus an epsilon for the high readiness/away period indices.
- Run an initial fitness run on this `S` and saving it in `n` for later printing.
- `P` is for use with `taskgroup` fitness functions, showing the time bar parameter.
- Rough calculations of the size of the space, the solution space, and the ratio between them.

Section: SA run

- Time start with `tic()`.
- Runs the simulated annealing algorithm and saves it in `out`.
- Time end saves in `solutionTime` with `toc()`.
- `S` rounded and final fitness value stored in `minima`.

- If `info?` is FALSE prints these values to the console.

Section: S clean up

- As done in Section: clean and print, `S` is cleaned up for easy printing.
- Final value of fitness saved.
- If `info?` is TRUE then final `S` is saved in the output file.

Section: improvement

- Three values now printed: the first fitness value, the final fitness value, and the ratio between them, where higher means a more improved fitness.

Section: final plot

- Similarly as Section: initial plot the name is created.
- The new diagnostic is saved in `dmf` this time.
- The plot is shown and saved if `pdf` is TRUE.
- Frees the output to the console if necessary.

A.2 File iterative.R

R Script *Last Updated: September 26, 2016*

Differences from main:

- Global start time is marked in `starttime` - useful for seeing how long the entire script runs.
- Folder name is created in the format `MMMDDtestprofiles` - automatically pulls the date, and will overwrite anything that is there.
- File name created with `output.txt` and `readme.txt` files.
- Ensures `pdf?`, `starttype`, `fitf`, and `manualbound` are appropriate for iteration.
- Begins writing data in the output file.
- This current version iterates over number of ships, specified in the for loop definition as 3:20 ships.
- The height of the pdf export depends on the number of ships; this is reassigned first.
- If there aren't enough ships for the task group fitness functions (`mptg` or `taskgroup`), an error is printed and the iterator will move onto the next number of ships.

- The iterator will show both the initial and final plots that are being saved, but since initial is on the screen longer (during the annealing run), it might not seem like both are being shown. This is useful for viewing completion progress of the iterator.
- After the iterations are complete, with the typical printed values into `output.txt`, the total final runtime is displayed in seconds and minutes, before closing the file.
- Opening the other file, `readme.txt`, the contents of the string `readmecontent` will be pasted. This should be changed before the entire file is run. This is the part that matters. Only change the lines with the dashes.
- Since this makes heavy use of `cat`, `pdf`, and `sink`, we do not recommend starting and stopping this function part way through often, otherwise outputs will end up in the wrong places. If this is necessary, make sure to type `ev.off()`and `sink()`.

Output files

- output.txt
- readme.txt
- plots

A.3 File importparameters.R

R Script *Last Updated: September 26, 2016*

Sheet: parameters

- The parameters sheet is read into `pars` as a data frame.
- Warning settings are saved before being turned off because accidental coercion errors are definitely here and no big deal.
- For each of the rows on the parameters sheet, lists, numbers, and strings are imported with their respective ways. Be sure for lists that `list` is in the `Units/type` column in the sheet, and the values themselves are separated with just a comma, no spaces—although commas and spaces might be fine, but this is untested.
- The old warning settings are reapplied.

Sheet: bound

- Left chart in sheet bound is imported as `bound`.
- Row names are assigned as the first column of bound.
- The matrix is melted to `longbound` (i.e. using the *melt* function in R) to make assigning easier.

- The variable name is created, in this case it is the `Min/Max/Ideal` (column name of `bound`, second column of `longbound`) followed by the `LTM/15/HR` (row name of `bound`, first column of `longbound`).
- The values are assigned to their variable name and set in the global environment.
- The direct vectors `mins` and `maxs` are assigned from the columns of `bound`.

Sheet: manualprofile

- First column is the state in number form; second column is the number of repetitions in the profile.
- If any ship state 8s appear anywhere in the manual profile, ship states 6 and 5 will be overlaid to shift, otherwise ship state 8 will do the shifting.
- Ship state 15s are used to ignore an open section of the profile from the shift area.
- This will only import if `manual?` is TRUE in the parameters sheet.
- Having every section in the manual profile the same length produces wild results. See `ex.pdf` for an example. It probably happens in the *rep*/*apply* section. It has been most likely fixed.

Sheet: rgb

- Just imports the 6th column of the rgb sheet, the rest can be changed without a problem.
- Change Function: `openstuff` if any more values are added to the initial profile.

Sheet: plotorder

- List of numbers in an order for the diagnostic plot.
- List of booleans for whether or not each colour should be shown.

A.4 File createMP.R

Function: createMP *Last Updated: September 26, 2016*

Input:

- ss = Step [positive integer]

Implicit:

- CycleLength [nonnegative integer]
- LTMLength [nonnegative integer]

- RRLength [nonnegative integer]
- STMLength [nonnegative integer]
- nSTMs [nonnegative integer]

Process:

- Normalizes `CycleLength`, `LTMLength`, `RRLength`, and `STMLength` by step size `ss` (`cl`, `ll`, `rl`, `sl`).
- Creates a `blank` profile `pr`.
- Finds the time that the profile will be left open, `timeop`, and subtracts the amount of time in all maintenance types from the total cycle length.
- Divides the total amount of time open by the amount of maintenance periods there are (`openlen`).
- Assigns the first section for long term maintenance (ship state 2), and the second for restricted readiness (ship state 3); creates a temporary time step `tt` to iterate over each of the short term maintenance periods (ship state 9).

Output:

- pr - vector with O&M profile
- `Lengths mismatch`

Function: awayblock *Last Updated: September 26, 2016*

Input:

- ss = Step

Implicit:

- CycleLength
- HRLength [nonnegative integer]

Process:

- Normalizes `CycleLength`, `HRLength` by step size `ss` (`cl`, `hl`).
- Initializes a `blank` profile (`pr`) of length `cl`.
- Assigns the first block as `away` for easier addition to a regular profile.

Output

- pr - vector starting with one block of 8s and the rest 0s

A.5 File forallopen.R

Function: openstuff *Last Updated: September 26, 2016*

Input:

- pr [O&M profile vector]

Implicit:

- used [vector of ship state numbers]

Process:

- Filters the values in `used` from the profile before starting, in order to keep uniformity.
- Finds the difference between consecutive values to best find the beginning of a block of a number. Since we're looking for the start and end of the new zero blocks, the vectors `zeros` and `rdszeros` are a list of indices where the value is equal to 0.
- `rdszeros` not currently used, but could come in handy when trying to find ready-duty ships that are not necessarily in high readiness state.
- `openstartdiff` calculated by finding the indices of the values that are lower than the value before it, typically indicating the beginning of a block of zeros.
  - There is a case that catches when the first value of a profile is also the beginning of a block of zeros, and when the last value of a profile is the end of a block of zeros.
- `openstarts` is the set intersection of `openstartdiff` and `nzeroes`, making sure that the negative value is only counted when the first of the block is a zero.
- `openenddiff` is similar as `openstartdiff` but uses positive values instead of negative values.
  - Similarly, there is a case that catches when the last value of a profile is the end of a block of zeros.
- `openends` is similar to `openstarts` but makes sure the index is one shifted left, since `openends` is technically the index of the first value not equal to zero following a block of zeros.
- The lengths of each open section is calculated for `openlens`
  - It should be noted that this will calculate in rare cases that `openstarts` and `openends` have different lengths, so long as the length of one is a multiple of the length of the other. Ignore it if this occurs (things are wrong).
- The three vectors, `openstarts`, `openends` and `openlens`, are stored in an atomized list and returned.

Output:

- a - atomized list
  - a$openstarts - vector of start indices for a block of 0s
  - a$openends - vector of end indices for a block of 0s
  - a$openlens - vector of lengths for each block of 0s

Function: awaystuff *Last Updated: September 26, 2016*

Input:

- pr

Implicit:

- halfmission [nonnegative integer]
- traveltime [nonnegative integer]
- HRLength
- Step
- Function: openstuff

Process:

- Using values calculated in `openstuff`, determines which blocks of zeros are long enough to host a high readiness block.
- If ship state 8 is already inputted directly to the (manual) profile, ship states 6 and 5 are used for the block that can be optimized, and `traveltime` and `halfmission` are used to determine length.
- Finds a vector `awayable` that is a vector of booleans for which of the blocks are able to host a high readiness block.
- Globally reassigns the new `nHRs` value to the number of TRUEs in `awayable`.
- Filters the `openends` by `awayable` and subtracts the length of the high readiness period to find when each of the high readiness blocks would best start, flush to the right edge of the open period.
- Calculates how far each open block (filtered by `awayable`) would allow the high readiness block to shift.
- Some of this terminology is due to change - deployed instead of high readiness, high readiness instead of away, etc.

- The three vectors, `awayable`, `awaystarts`, and `shiftable` are stored in an atomized list and returned.

Output:

- a - atomized list
  - a$awayable - vector of booleans for which openlens can fit an HR section.
  - a$awaystarts - vector of start periods of openstarts filtered by away-able
  - a$shiftable - vector of lengths of shift allowable in each open section.

A.6 File preplot.R

Function: preplot *Last Updated: September 26, 2016*

Input:

- S [vector of state starts]
- ss = Step
- pr = createMP(ss)

Implicit:

- CycleLength [nonnegative integer]
- HRLength [nonnegative integer]
- halfmission [nonnegative integer]
- traveltime [nonnegative integer]
- AWAY [boolean]
- nShips [positive integer]
- nHRs [nonnegative integer]
- Function: circshift

Process:

- Normalizes `CycleLength`, `HRLength` by step size `ss` (`cl`, `hl`)
- Creates a blank data frame with one extra time step.
- If not including away period
  - For each ship `n`
    - starts the profile at `S[n]` with `circshift`

        * Names the column in the data frame `n`

- If including away period
  - If ship state 8 is already inside the manual profile assignment, uses ship states 6 and 5 instead for the value that will be shifting, and changes the block length to depend on `halfmission` and `traveltime` instead of `HRLength`.
  - If `allopen` is TRUE, each open section with enough space for a high readiness block will be populated, so the number of high readiness blocks is changed to the number of open sections that can fit them. This is also globally done in Function away-stuff.
  - Subsets the entire length of `S` into the number of sections per ship `subslen`.
  - For each ship `n`
    * Subsets the `S` into the piece `subS` belonging to ship `n`.
    * Creates a duplicate profile `pr2` to edit.
    * Reassigns the appropriate value to each position specified in `subS` (except the first) - ship state 8, or ship states 6 and 5.
    * Adds an empty timestamps to the end of the appropriately shifted profile `pr2` to find the finished profile and add it to the data frame.

Output:

- a - atomized list
  - a$fleet - data frame (ncol = nShips, nrow = nPeriods+1)

A.7 File diagmat.R

Function: diagmat *Last Updated: September 26, 2016*

Input:

- fleetin [list containing data frame (ncol = nShips, nrow = nPeriods+1) at $fleet]
- ss = Step

Implicit:

- CycleLength
- colorder [vector of ship states]
- green? [boolean]
- inplot [vector of booleans]

Process:

- Extracts the frame `df` from `fleetin$fleet`.
- Removes the first column, the time column, for easier iteration (`tdf`).
- Normalizes `CycleLength` by step size `ss` (`cl`).
- Creates an output data frame at `outd` (with one extra time step).
- Creates new colour order vector `colourorder` from `colorder` and what is marked as TRUE in `inplot`.
- For each colour `n` in new `colourorder`:
  - Collects how many ships at each timestep is equal to ship state `n`.
  - Inputs into data frame `outd` with name `col n`.
- For instances where there are colours hidden, in order to avoid having blank spaces at the top, there is a column called `extra` with that removed amount.
- That extra column is grouped with the white/light green and not above the dark green as it would be by default.
- `outd` is returned with the original schedule data frame `df` in list `a`.

Output:

- a - atomized list
  - a$fleet: (ncol = nShips, nrow = nPeriods+1).
  - a$diagnostic: (ncol = ncolours+1, nrow = nPeriods+1).

A.8 File multiplot.R

Function: multiplot *Last Updated: September 26, 2016*

(see `http://www.cookbook-r.com/Graphs/Multiple_graphs_on_one_page_(ggplot2)` for details on how the function *multiplot.R* was created.)

Input:

- plotlist = NULL [ggplots]
- cols = 1 [positive integer]
- layout = NULL [matrix]

Implicit:

- NA

Process:

- Imports `grid`.
- Determines number of plots by the length of the rest of things and the `plotlist`.
- If there is a `layout` present, uses that arrangement.
- If there is no layout, uses the `cols` value to determine how many columns.
- Else just plots automatically.

Output:

- Multiple plots in one dev window.

A.9 File plotfleet.R

Function: plotfleet *Last Updated: September 29, 2016*

Input:

- fleetin [list with data frame (ncol = nShips, nrow = nPeriods+1) at $fleet and data frame (ncol = ncolours+1, nrow = nPeriods+1) at $diagnostic]
- ss = Step
- both = `both?` [boolean]

Implicit:

- CycleLength
- tickStep [positive integer]
- `green?`
- qcol [vector of ship states]
- inplot
- Function: multiplot

Process:

- Normalizes `CycleLength` and `tickStep` by step size `ss`.
- Creates the amount of gap `lht` between each line of the main ship schedule, depending on whether or not the background is white or green.
- Extracts frame `df` from `fleet$fleetin`.

- Melts `df` by time and creates axis vectors as factors, numeric, and continuous values.
- Creates lists for the axis breaks.
- plot 1 (can be called individually with `p1`)
  - ggplot basics.
  - geom_rect specifications for `xmin`, `xmax` (`xmin+1`), `ymin` (reversed by `nShips-values`, centered by adding 0.5, and shrunk for line spacing with `lht`), `ymax` (`ymin+1`, reversed with `nShips-values`, centered by adding 0.5, shrunk by line spacing `lht`).
  - axis titles.
  - breaklists and axis labels.
  - manual scale colouring, specified by which index of `qcol`, which is a function of `inplot` and `green?`
  - panel specifications, particularly grid lines.
  - plot edge space.
  - axis text colour (might not be necessary—will test later).
- if `both?` is TRUE, plot 2 (can be called individually with `p2`)
  - First double checks if there is a second data frame to plot at `fleetin$diagnostic`, if not then just plots the first automatically.
  - Same process as plot 1 except `geom_bar` instead of `geom_rect`, and with no line spacing.

Output:

- p1
- p2
- multiplot(p1,p2)

A.10 File conversions.R

Function: m2d *Last Updated: September 26, 2016*

Input:

- matt [matrix (nrow = nPeriods, ncol = nShips)]

Implicit:

- NA

Process:

- Transposes matrix and adds an extra timestep to the end (`d2`).
- Adds another time column to the first spot in the matrix (`d3`).
- Changes to data frame `d4`.
- Gives names to the columns and rows.
- List `a` of one element callable at `$fleet`.

Output:

- a$fleet - data frame (nrow = nPeriods + 1, ncol = nShips + 1) containing the schedule converted from matrix form
- opposite of Function: d2m

Function: d2m *Last Updated: September 26, 2016*

Input:

- dff (list containing one data frame (nrow = nPeriods + 1, ncol = nShips + 1) at $fleet)

Implicit:

- NA

Process:

- Pulls the data frame from the list (`p`).
- Removes last row and first column, transposes, and returns.

Output:

- matrix (nrow = nPeriods, ncol = nShips) containing schedule converted from data frame
- opposite of Function: m2d

A.11 File idealbound.R

Function: idealbound *Last Updated: September 26, 2016*

Input:

- NA

Implicit:

- manual [boolean]
- profile
- Step
- nSTMs
- allopen [boolean]
- nShips
- LTMLength
- STMLength
- RRLength
- HRLength
- nHRs
- CycleLength
- Function: awaystuff
- mins, maxs [vectors of nonnegative integers]
  - MaxTotal, MinTotal
  - MaxLTM, MinLTM
  - MaxRR, MinRR
  - MaxSTM, MinSTM
  - MaxHR, MinHR

Process:

- If using a manual profile, `LTMLength`, `nSTMs`, `nHRs`, `CycleLength`, `RRLength`, and `STMLength` are recalculated and globally reassigned.
- `mins` for `-total`, `-LTM`, `-RR`, `-STM`, and `-HR` are calculated with floor of `-Length`, and `maxs` with ceiling, and globally assigned both into the `mins`/`maxs` vectors and into the individually imported `MinTotal`, etc.

Output:

- NA

A.12 File fitness.R

Function: mcount *Last Updated: September 27, 2016*

Input:

- S
- prf [O&M profile, vector of ship states]

Implicit:

- mins, maxs
  - MaxTotal, MinTotal
  - MaxLTM, MinLTM
  - MaxSTM, MinSTM
- colorder
- Function: d2m

Process:

- Converts data frame to matrix `sch` with `d2m`.
- Creates diagnostic vectors
  - `overlap` makes sure no values are above the maximum possible (max(colorder)).
  - `somme2` is the amount of ship states that are 2 in a given timestep.
  - `somme9` is the amount of ship states that are 9 in a given timestep.
  - `sommeTot` is the sum of both `somme2` and `somme9`.
- Counts how many from each diagnostic vector is above the minimum or maximum allowed for each ship state, weights and adds them together for an integer valued `errorsum`

Output:

- errorsum - numeric value as a grading the schedule

Function: HRcount *Last Updated: September 27, 2016*

Input:

- S
- prf [O&M profile, vector of ship states]

Implicit:

- mins, maxs
  - MaxHR, MinHR
- Function: d2m

Process:

- Converts data frame to matrix `sch` with `d2m`.
- Creates diagnostic vector
  - `somme8` is the amount of ship states that are 8, 6, or 5 in a given timestep.
- Counts how many from each diagnostic vector is above the minimum or maximum allowed for each ship states 8, 6, or 5, valued `errorsum`

Output:

- errorsum - numeric value as a grading the schedule

Function: taskgroup *Last Updated: September 30, 2016*

Input:

- S
- prf
- ind = 1

Implicit:

- used
- halfmission
- traveltime
- Step
- MinMission
- RDSboth
- nPeriods [positive integer]
- nRDSs [nonnegative integer]
- nShips
- nROTO [0 or 1]

- Function: d2m
- Function: preplot

Process:

- Converts the state vector `S` and O&M profile `prf` to a schedule `sch` with `d2m` and `preplot`.
- If 1, 5, 6, and 8 are ship states in `used`, then they are filtered out to all equal 0.
- The amount of open ships in a given timestep is stored in `nzeroes`.
- Scales `halfmission` and `traveltime` by `Step`.
- Vector `comparisonvec` created for later use with `durationmatrix` to compare for ready duty ship availability over the duration of a task group deployment. Currently not flexible for various traveltime overlap situations.
- Creating a simplified `half time` variable called `len1` and a `full time` variable called `len2` (this is where travel overlap or not can be changed for flexibility).
- Initializes blank lists of lists, `whichopen`, `now`, `noworthen`, `rdsopen`, `rdsonly`, and `rdsduring`.
- Finds which ships in each `timestep` are open and stores in `whichopen`.
- For each timestep, if the number of zeros is less than what is needed to send an immediate task group, that section is set to blank (might need to be changed), else if a ship is available for the duration (`halfmission`+2 $\times$ `traveltime`), then the ships are listed in the `now` vector.
- A similar list, `rdsopen`, includes values of ship states 7 and 15, instead of just what was changed to ship state 0 at the beginning of the function. (this is where the setting is for `RDSboth`)
- The `then` vector is a circular shifted `now` vector, by the length of the `halfmission`.
- Another vector `noworthen` is the union of ships available to be sent either `now` or `then`.
- The number of ships at each timestep in `noworthen` is stored in `countnoworthen` to make quantification for the set. Similarly with `countrdsonly` for `rdsdonly`.
- `v1true` is for which timesteps a task group only is possible.
- `durationmatrix` is a matrix (nrow = mlen, ncol = nPeriods) whose first row is number of ships available at that time and where each row is the previous row shifted by one. `v2true` is checking that for each column, every value is greater than `comparisonvec`, checking the equivalent of each `mlen` slice of `rdsrightnow` `v1` is the number of timesteps where a task group is possible.
- `v2` is the number of timesteps where having ready duty ships for the duration of the task group is possible.

- `resvec` is the combination of both as a resultant vector to be returned if `ind` is set to 2.
- Subtracting `v1` from `nPeriods` gives a minimizable result for the optimization run, and the number of timesteps where the task group is not possible.

Output:

- Number of timesteps where a task group would not be able to be sent.

Function: mptg *Last Updated: September 26, 2016*

Input:

- S
- prf
- wts = c(1000000,1000,1) [vector of numbers as weights for the combination of the three fitness functions]

Implicit:

- Function: mcount
- Function: HRcount
- Function: taskgroup

Process:

- Runs functions `mcount`, `HRcount`, and `taskgroup`, and adds them with the weights `wts`.

Output:

- Value of the sum of functions weighted by `wts`.

Function: taskwrap *Last Updated: September 29, 2016*

Input:

- l [list]

Implicit:

- NA

Process:

- Returns the first of the input list.
- Depreciated

Output:

- First element of list `l`

Function: circshift *Last Updated: September 29, 2016*

Input:

- x [vector]
- n = 1 [positive integer]

Implicit:

- NA

Process:

- If `n` is 0, then the vector is returned identically.
- `n` is modulated to be less than the length of vector `x`.
- Otherwise, the last `n` elements of vector `x` are appended before the remaining elements of vector `x`, resulting in the first element now at the `n+1`th position, having been shifted right `n` places.
- There is a similar function `revcircshift` for use only within apply functions when the iterations are over the number of shifted indices.

Output:

- A shifted vector to the right by `n` places

Function: whichwrap *Last Updated: September 13, 2016*

Input:

- x [vector]
- a = 0 [integer]

Implicit:

- NA

Process:

- Returns the indices of elements of `x` that are equal to `a`, for easier use of `which` in an apply function.

Output:

- A vector of indices of `x`.

A.13 Additional notes

AWAY: Since there are three ways the high readiness/away process can be turned off, this is making sure all conditions are met in order to start running features related to the dark green/high readiness period, namely that the actual setting `Away?` is turned on (TRUE), the number of high readiness periods is specified to be greater than 0 (`nHRs`), and the length of the vector of starting positions for high readiness periods (`AwayStarts`) should be greater than zero (and hopefully equal to `nHRs`).

Profiles: Added in some quick profiles to avoid having to repetitively type them in to the manual profile box. Change 8s to 0s if you want dark green instead of blues, and set profile equal to each. Don't run the first few lines of Section: initial S and profile setup in *main.R*.

Annex B Input parameters

This annex lists the input parameters that are required to run the toolset. Note that the input parameters are specified through a Microsoft Excel Spreadsheet. In total, there are five worksheets in the spreadsheet—a screenshot of each of them is provided below to give an overview of the type of input parameters that are required.

B.1 Worksheet `parameters`—The main parameters

The main parameters that drive the toolset are shown in Figure B.1.

Input Parameter	Value	Units/type	Description
nShips	13	*ships*	number of ships in the fleet
CycleLength	3600	*days*	total length of maintenance cycle in days
Step	30	*days/period*	number of days per time step
nPeriods	120	*periods*	calculated value of the number of periods in the entire cycle
starttype	z	*{e,z,r}*	which initial state vector should be used: equally spacing the ships, starting them all from zero, or a random start time
Away?	FALSE	*boolean*	whether or not the S vector is just LTM start dates (F) or includes 'away' period start dates (T)
manual?	TRUE	*boolean*	whether or not the maintenance profile is created automatically (F) or is inputted manually on sheet:manualprofile (T)
manualbound	TRUE	*boolean*	for manual maintenance profiles, whether or not the boundaries should be automatically calculated (F) or imported in sheet:bound (T)
allopen	FALSE	*boolean*	specifies if each open gap that can fit an HR section should have an HR (T), or if just the first nHRs should be used instead (F)
LTMLength	360	*days*	length of a long term maintenance period
STMLength	120	*days*	length of a short term maintenance period
nSTMs	3	*number*	number of short term maintenance periods in cycle
RRLength	0	*days*	length of the restricted readiness period
HRLength	180	*days*	length of time the high readiness or 'away' period lasts
nHRs	0	*number*	number of high readiness or 'away' periods to use if allopen is false (should match length of AwayStarts)
AwayStarts	16,28,40,52	*list*	list of start dates for HR/Away periods, relative to the start of LTM
maxshiftrange	24	*number*	the number of timesteps to the left the awaystarts can shift - calculated value, do not change
shiftrange	6	*number*	the number of timesteps to the left that we LET the awaystarts shift, must be less than maxshiftrange
TimeOut	1	*minutes*	length of time the simulated annealing algorithm should run before quitting
halfmission	180	*days*	number of timesteps of a 'half mission'
traveltime	30	*days*	number of timesteps that the halfmissions would need for travel time
MinMission	4	*ships*	number of ships needed for a halfmission
nRDSs	2	*ships*	number of ships needed for ready duty that aren't being used in the mission
nROTO	1	*rotations*	1 for 1 switch, 0 for 0 switches, untested for higher
used	0,8,6,5	*list*	which colours an away section can be found in
RDSboth	TRUE	*boolean*	if ready duty ship should be strictly only in the HR/dark green (F) or if it can be included in the NR/light green (T)

Figure B.1: *Input worksheet—`parameters` (Part I).*

tickStep	360	*days*	for use in the plot
both?	TRUE	*boolean*	plot export parameter, if both the schedule and the histogram are both wanted (T), or if just the schedule (F)
green?	TRUE	*boolean*	if the empty sections in the schedule and diagnostic plot should be green (T) or white (F)
ww	8.5	*inches*	width of pdf export
hh	9.34	*inches*	height of pdf export, calculated value
fname	temp	*string*	unique filename for the pdf output to start with
pdf?	TRUE	*boolean*	setting for whether or not a PDF should be exported (T) or just plotted into the console (F)
info?	FALSE	*boolean*	setting for whether or not a output parameters should be exported (T) or not created at all and output is left to print into the console (F)

Figure B.1: *Input worksheet—`parameters` (Part II).*

B.2 Worksheet `rgb`—Possible platform status and associated RGB colors

The worksheet in Figure B.2 is used to specify the possible status that the platforms can be at in the OPCYCLE. Note that in its current implementation, it is limited to 15 status values.

V	Type Name	R	G	B	Color	R	G	B
1	Unassigned	1.000	1.000	1.000	white	255	255	255
2	ER-LTM	1.000	0.000	0.000	red	255	0	0
3	RR	0.000	0.000	0.000	burlywood2	0	0	0
4	Post-deploy	1.000	0.949	0.000	yellow1	255	242	0
5	Transit	0.741	0.871	1.000	skyblue1	189	222	255
6	On Station	0.118	0.553	0.910	dodgerblue2	30	141	232
7	SR	0.596	0.984	0.596	palegreen	152	251	152
8	HR	0.180	0.545	0.341	seagreen	46	139	87
9	ER-STM	0.631	0.027	0.604	darkmagenta	161	7	154
10	PersTempo	1.000	0.514	1.000	orchid1	255	131	255
11	ECWP				skyblue1			
12	blank2				white	247	150	70
13	RR	0.980	0.749	0.561	tan1	250	191	143
14	blank4				white			
15	ignored	0.000	0.000	0.000	black	0	0	0

Figure B.2: *Input worksheet—`rgb`.*

B.3 Worksheet `manualprofile`

When running in `manualprofile` mode (see Row 8 in Figure B.1), the user can specify to profile of interest using the worksheet presented in Figure B.3.

type	amount
2	360
7	360
8	360
9	120
7	360
8	360
9	120
7	360
8	360
9	120
7	360
8	360

3600

Figure B.3: *Input worksheet—`manualprofile`.*

B.4 Worksheet `bound`—Setting individual platform status lower and upper bound

Using the worksheet in Figure B.4, the user can set the acceptable bounds for each platform status in the OPCYCLE. These values can be used in the fitness functions as defined by the user.

bound	Min	Max
Total	4	5
Open	0	0
LTM	1	2
RR	-1000	1000
4	0	0
5	0	0
6	0	0
SR	0	0
HR	5	6
STM	0	0
10	0	0
11	0	0
12	0	0
13	0	0
14	0	0
15	0	0

Figure B.4: *Input worksheet—`bound`.*

B.5 Worksheet `plotorder`

A screenshot of the `plotorder` sheet is provided in Figure B.5. It is used to show which platform status (i.e. SR, HR, ..., as per Worksheet `rgb`) are displayed in the figure and in what order.

8	TRUE
6	TRUE
5	TRUE
0	TRUE
1	TRUE
7	TRUE
4	TRUE
11	FALSE
12	FALSE
13	FALSE
14	FALSE
15	FALSE
3	TRUE
10	TRUE
9	TRUE
2	TRUE

Figure B.5: *Input worksheet —`plotorder`.*

List of acronyms

CORA	Centre for Operational Research and Analysis
DGNFD	Director General Naval Force Development
DMP	Deep Maintenance Period
DRDC	Defence Research and Development Canada
ER	Extended Readiness
FSWEP	Federal Student Work Experience Program
HR	High Readiness
LTM	Long-Term Maintenance
MORT	Maritime Operational Research Team
NR	Normal Readiness
O&M	Operations and Maintenance
OPCYCLE	Operational Cycle
RCN	Royal Canadian Navy
RD	Reference Document
RDS	Ready Duty Ship
RR	Restricted Readiness
SMP	Shallow Maintenance Period
SR	Standard Readiness
STM	Short Term Maintenance
TG	Task Group
TRP	Tiered Readiness Programme

DOCUMENT CONTROL DATA

(Security markings for the title, abstract and indexing annotation must be entered when the document is Classified or Protected.)

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.)

DRDC – Centre for Operational Research and Analysis
Dept. of National Defence, MGen G. R. Pearkes Bldg., 101 Colonel By Drive, 6CBS, Ottawa
ON K1A 0K2, Canada

2a. SECURITY MARKING (Overall security marking of the document, including supplemental markings if applicable.)

UNCLASSIFIED

2b. CONTROLLED GOODS

(NON-CONTROLLED GOODS)
DMC A
REVIEW: GCEC DECEMBER 2013

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.)

On the use of simulated annealing to optimize maintenance schedules for naval platforms

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

Green, M.; Caron, J.-D.; Fong, V.

5. DATE OF PUBLICATION (Month and year of publication of document.)	6a. NO. OF PAGES (Total containing information. Include Annexes, Appendices, etc.)	6b. NO. OF REFS (Total cited in document.)
December 29, 2016	57	12

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.)

Reference Document

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

DRDC – Centre for Operational Research and Analysis
Dept. of National Defence, MGen G. R. Pearkes Bldg., 101 Colonel By Drive, 6CBS, 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.)	9b. CONTRACT NO. (If appropriate, the applicable number under which the document was written.)
01aa	

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.)	10b. OTHER DOCUMENT NO(s). (Any other numbers which may be assigned this document either by the originator or by the sponsor.)
DRDC-RDDC-2016-D084	

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

Unlimited

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.)

In the summer of 2016, the MORT hired a student through the FSWEP to develop a toolset for generating fleet availability schedules. The blocks of availability include both Standard (or Normal) and High Readiness periods while ships are considered unavailable when in maintenance (long or short term periods) and while in the tiered-readiness program. The toolset, created in R, uses simulated annealing optimization technique. Some of the toolset input parameters include the fleet size and the Operations and Maintenance (O&M) profile of interest for the fleet. This document serves as a summary of the work that was accomplished during the 10-week period. It describes the motivation, defines the problem, and provides some implementation details which are illustrated with a simple example.

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.)

Royal Canadian Navy; RCN; Maintenance; Schedule; Simulated Annealing; Optimization; Heuristic; R

DRDC | RDDC

SCIENCE, TECHNOLOGY AND KNOWLEDGE FOR CANADA'S DEFENCE AND SECURITY | SCIENCE, TECHNOLOGIE ET SAVOIR POUR LA DÉFENSE ET LA SÉCURITÉ DU CANADA

www.drdc-rddc.gc.ca