Optimising Maintenance Intervals for a Component using a New Hill-Climbing Algorithm

Shawulu Hunira Nggada

Department of Computer Science, University of Hull, HU6 7RX, UK
shnggada@googlemail.com, S.H.Nggada@2007.hull.ac.uk

Abstract

Maintenance reduces the occurrence of failures of a component (or IT product) and also extends its useful life. This way the time to dispose of such a component may be delayed and this reduces the rate at which greenhouse gases enter into the food chain and the atmosphere. It will be helpful to produce set of maintenance intervals for such component at which its extended useful life and cost are most improved. This set will consist of trade-offs between extended life and cost and is known as optimal set. This paper builds on earlier work to define the optimisation of the maintenance intervals of a given component with respect to its extended useful life and cost. The paper also establishes and uses a new approach of hill-climbing search algorithm.

Keywords: *Green computing, optimization, hill-climbing, preventive maintenance, eco-friendly*

1. Introduction

There has been call for changes in the manufacturing culture following the introduction of international and national legislations on industrial production and waste management [1]. The regulations suggest that equipment manufacturers are responsible for the disposal of their manufactured products. This infers that manufacturers are required to incorporate in the design of a product (or component) how such component will be disposed. The disposal could either be in the form of recycling or properly getting rid of the component. Since not all parts of a component may be recyclable, it will therefore be useful if the time to dispose is extended. This suggests that the rate of disposal would be reduced. An earlier work [2] on green computing showed that this can be achieved through maintenance. In particular, an interval based maintenance also referred to as preventive maintenance was considered. It was assumed that the maintenance action does not make the component to as good as new and hence imperfect preventive maintenance policy was considered as it is in this paper.

In considering the disposal of component from its early design stage, manufacturers can consider the maintenance interval for such component that would improve both its extended useful life and cost. This is exactly what this paper addresses; to investigate the possibility of suggesting such maintenance interval which could be considered from early design stage. It is also possible to generate set of maintenance intervals for which none is better than any other member of the set. This set contains maintenance intervals that are non-dominated or simply trade-offs between extended useful life and cost. In optimisation terms the extended useful life and cost are referred to as objective functions. If *x* and *y* are two maintenance intervals where *x* dominates *y*, then this implies that *x* is not inferior to *y* in any of the objective functions but *x* is superior to *y* in at least one of the objective functions. Component maintenance can be performed at different time intervals, for instance 10, 30, 50, etc. The problem that is specifically addressed in this paper is what subset of the available time options if used will improve the component's extended useful life and cost.

In section 2 the concept of potential solution space is presented. Section 3 presents three search algorithms that are common in literature and are used in the search for solution or solutions within a space of potential solutions. In particular, simulated annealing, genetic algorithm and hill-climbing are briefly discussed. In section 4, a variant of the hill-climbing algorithm called steady surface hill-climbing (SSHC) is established. The SSHC is used in optimising the maintenance intervals of a component. In section 5, results and evaluation are presented while section 6 draws conclusions.

2. Optimisation Spaces

Optimisation is concerned with searching and finding a better solution or better solutions within a defined population of potential solutions. The defined population is termed as solution space and a better solution as optimal solution. Where the optimisation returns a set of better solutions then the set is referred to as optimal set. Within a defined solution space, there exists a population of solutions that are potentially feasible. This population is referred to as feasible region. Figure 1 depicts this relationship.

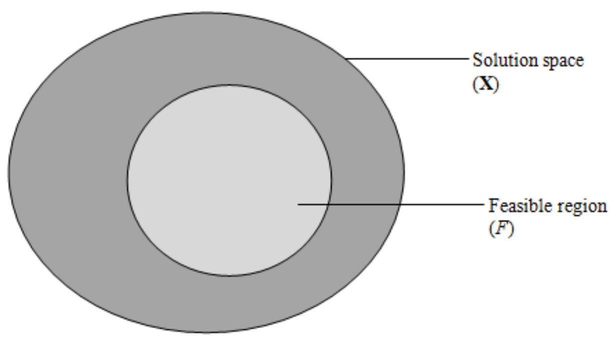


Figure 1. Relationship between Feasible Region and Solution Space

If $\mathbf{X}$ denotes the solution space and F the feasible region, then the relationship is defined as $F \subset \mathbf{X}$. Optimisation is not new, however the process is often manual, time consuming and involves a step by step approach to identify the right combination of the associated process parameters for the best solution. The manual approach does not allow a thorough exploration of the solution space [3] and makes it difficult to enumerate potentially feasible solutions which would quickly guide the search towards optimal or near optimal solutions. A process of automation in optimisation is therefore helpful to reduce design time and to also improve the design.

3. Search Algorithms

A search algorithm in optimisation terms is one that can explore the population of potential solutions for a given optimisation problem and to finding those solutions that are optimal or near optimal. An optimisation that has more than one objective is termed as multi-objective, such as the problem in this paper where extended useful life and cost form the objectives. Some of the existing search techniques are suitable for single objective while others for multi-objective. An optimisation problem can converge prematurely to a local optimum or efficiently to a global optimum. A local optimum consists of solutions that are better than their near neighbours but may be inferior to others away from them though within the same solution space. In contrast, a global optimum consists of those solutions that are globally better within the given solution space.

This section briefly introduces three search algorithms as mentioned earlier; genetic algorithm, simulated annealing and hill-climbing. More focus and emphasis is however given to hill-climbing being the choice for the problem in this paper. The proposed new hill-climbing technique "steady surface hill-climbing" (SSHC) is established in section 4.

3.1. Genetic Algorithm

Genetic algorithm (GA) is a search algorithm that is based on the mechanics of natural selection and natural genetics [4]. It is widely used in engineering applications and it is suitable for a multi-objective optimisation problem. GA is also efficient and effective for an optimisation problem with huge solution space. It is also known to converge to a global optimum. GA defines a mapping between the problem (for instance a system model being optimized) and its digital representation which can be easily manipulated. The digital representation can either be in the form of binary or integer strings and is termed as encoding. GA begins by randomly generating potential solutions (encodings) within the feasible region guided by defined constraints to the optimisation problem. Hence in GA, the feasible region is defined by the constraints imposed on the optimisation problem. The fitness of every solution in GA is evaluated according to the objective functions of the optimisation. GA uses the notion of "survival of the fittest" and over time it promotes fittest solutions to passing their traits unto the next generation. It also defines move operators such as crossover and mutation. In crossover, two parents are randomly selected for recombination (breeding) to produce an off-spring while mutation injects a new trait into the off-spring in the hope of infusing diversity into the population. The implementation of genetic algorithm is not easy but it is worth in addressing complex optimisation problems.

More on the use of GA in solving specific optimisation problems in engineering can be found in [5] and [6]. For instance [5] addresses the optimisation of manufacturing process planning for multiple parts by modifying the traditional GA approach while [6] addresses the optimisation of preventive maintenance scheduling of a system by modifying the selection technique of GA.

3.2. Simulated Annealing

Simulated annealing is a search technique that is based on the annealing process as used in thermodynamics [7]. In order to prevent the likelihood for premature convergence, simulated annealing allows move to inferior solutions especially at early stages of the optimisation. For instance when a metal is heated at high temperature, it could be molded to assume any desired shape. As its temperature cools down, it becomes stiffer. Simulated annealing uses this analogy in its optimisation process. At the start of optimisation, simulated annealing sets a temperature value and the method through which it cools. The process is flexible at the early stages and accepting poor quality solutions with the aim that the solutions become better as the temperature cools down. The acceptance of solutions becomes more strict or rigid as the temperature cools. Simulated annealing is known to be suitable for single objective optimisation problems; however other variants of it such as the classical simulated annealing based multi-objective algorithm can address multi-objective optimisation problems [8]. More on simulated annealing can be found in [7] and [8].

3.3. Hill-climbing

The hill-climbing search technique is analogous to the act of climbing a hill. Within a hilly environment there may exist several hill tops. The aim of hill-climbing is to attain the hill-top of the highest hill within the given environment. If a climber begins from a point say P_i, the climber's next or new position will be a point P_{i+1}. The new position is accepted if it is better than the current. This process continues until positions are no longer found to be better than the current. Figure 2 is an illustration of the concept of hill-climbing.

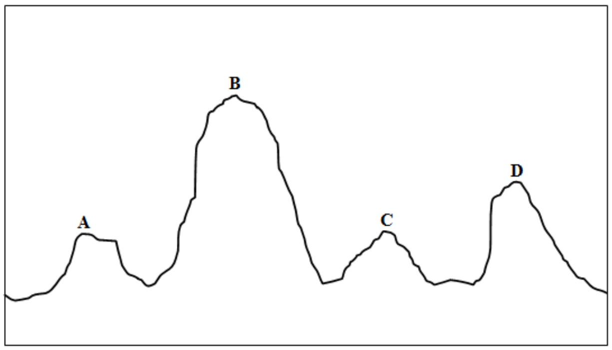


Figure 2. The Concept of Hill-climbing

Where: **A**, **C** and **D** are local optima
B is a global optimum

In Figure 2, three local optima are indicated (**A**, **C** and **D**) and this infers that it is possible for the search to get stuck in one of these optima and to present the solution at that optimum as the optimal. One way out of the problem of local optima is to use iterated hill-climbing. In iterated hill-climbing, the hill-climber is restarted with a different initial start position (P_i) when a local optimum has or is suspected to have been found [7]. One challenge however is in defining the method by which a convergence to a local optimum can be detected. An elaborate form of the traditional hill-climbing algorithm found in [9] is described below. This algorithm will be used to establish the steady surface hill-climbing algorithm.

(i) Pick a solution S_c from the search space **X**

(ii) Evaluate S_c

(iii) Select a solution S_n in the neighbourhood of S_c

(iv) Evaluate S_n

(v) If S_n is better than S_c then $S_c = S_n$, else discard S_n

(vi) Return to step ii until stop criterion applies

4. Steady Surface Hill-climbing (SSHC) Technique

Traditional hill-climbing algorithm is suited for single objective optimisation problem and hence returns a single optimal solution. This paper proposes a new approach to the hill climbing algorithm using a simple concept. Figure 3 below is similar to Figure 2 except that the highest hill-top **R** in Figure 3 is a plateau (leveled top) otherwise known as steady surface.

The concept of the steady surface is that solutions exist on such surface which themselves are not better than each other but are all better than other solutions on hill-tops elsewhere within the environment. In simple terms the plateau or steady surface consists of non-dominated solutions. The aim of the new hill-climbing algorithm "steady surface hill-climbing" is to find this plateau (or the highest plateau) and to return the set of optimal solutions it contains.

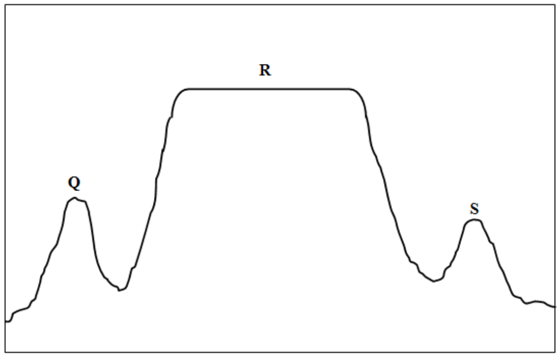


Figure 3. The Concept of Steady Surface Hill-climbing (SSHC)

Where: **Q** and **S** are examples of local optima
R is a global optimum (optimal set)

The steady surface hill-climbing algorithm attempts to exhaustively cover the environment depending on the value of the move parameter. This *move* parameter is iteratively used to progress to subsequent positions by adding it to the *next* parameter. The algorithm uses these two parameters to generate the population P of potential solutions for the optimisation. Each solution in the population is evaluated for fitness, i.e. against the set objective functions of the optimisation. The solutions in the population are then compared for domination. A found non-dominated solution is then added to the archive population P_A. The final result is a set of non-dominated solutions contained in the archive population P_A. Based on the traditional hill-climbing algorithm, the steady surface hill-climbing algorithm is presented below.

(i) Set the value of the *move* parameter

(ii) Set next move parameter to zero; $next = 0$

(iii) Set move counter; $n = 0$

(iv) Set population P to null; $P = \emptyset$

(v) Initialize archive population P_A to null; $P_A = \emptyset$

(vi) To generate the population P, Repeat steps *a* to *c* below until the environment is covered

(a) $next = next + move$

(b) $P = P \cup next$

(c) $n = n + 1$

(vii) $\forall p \in P$, evaluate p for fitness

(viii) $\vee_{j=1}^{n}(p_j \in P)$ do steps *d* to *f* below

(d) Set counter $k = 0$

(e) Set $q = p_j$

(f) Repeat steps *g* to *i* until $k = n$ *or* q *is dominated*

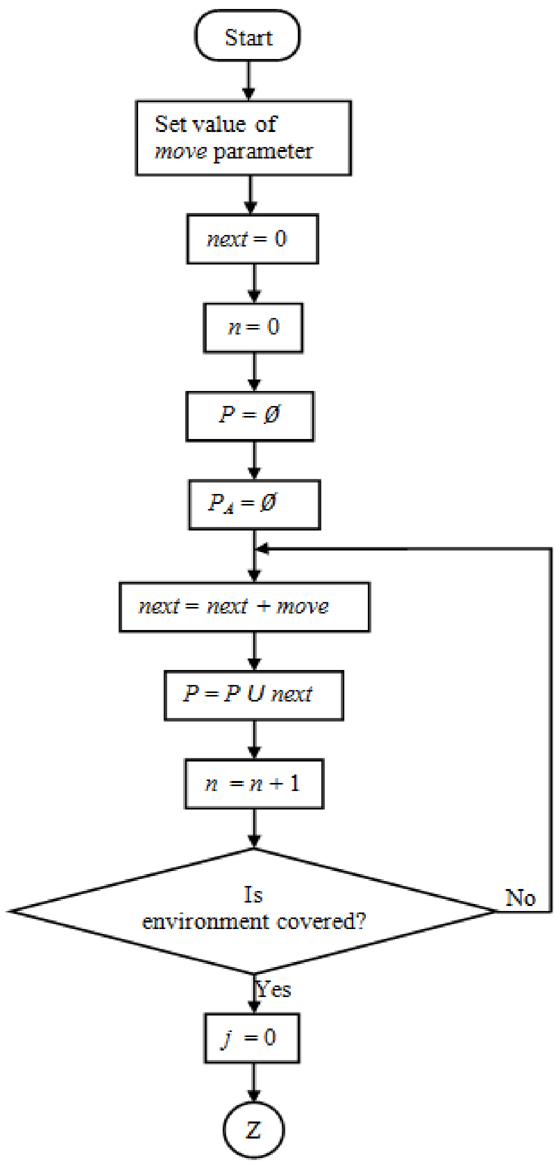
(g) $k = k + 1$

(h) If q is dominated by p_k then go to step *viii*

(i) *If* $k = n$ and q is not dominated then $P_A = P_A \cup q$

(ix) Return the non-dominated set in the archive population P_A as optimal solutions

Figure 4 is a simple and detailed flowchart which gives a visual representation of the above algorithm.



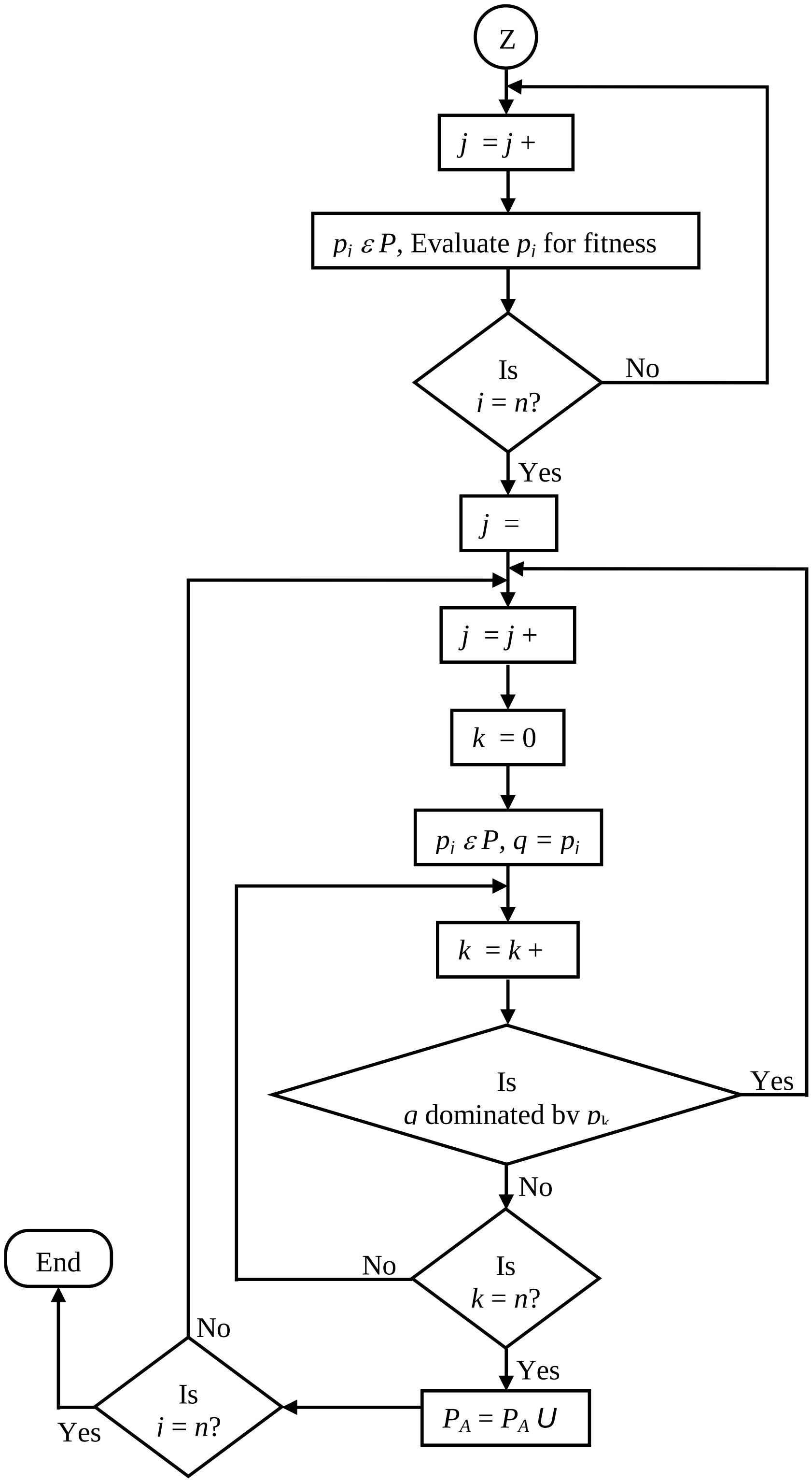


Figure 4. Flowchart of SSHC

Having established the steady surface hill-climbing algorithm, the optimisation problem for the maintenance intervals can now be defined as follows.

$\mathbf{F}(T_p) = \{ \max[L_N(T_p)], \min[C_T(T_p)] \}$

such that:

$T_p \in \mathbf{X}$,

Where: L_N is the extended useful life of the component
C_T is the total component cost per unit time under maintenance
T_p is maintenance interval
X is the solution space
F is a function that returns an optimal solution

The optimisation problem pursues the maximization of the component's extended life and the minimization of its total cost per unit time.

5. Results

As this paper is based on an earlier work found in [2], same evaluation models are here used for component extended useful life and cost. These models are here presented in equations 1 and 2 respectively. The optimisation was performed under the assumption that the effectiveness of the maintenance actions depreciates as the component ages, implying a varying improvement factor. The model used for varying improvement factor as found in [2] is presented in equation 3. It is also assumed that component failure follows the Weibull distribution.

$$L_N = L + \sum_{j=1}^{n} f_j T_p \quad (1)$$

Where: L is the useful life of the component
f_j is the improvement factor at the *j-th* maintenance time
L_N and T_p are as defined in the previous section

$$C_T = \frac{\left(C_c + nC_{ipm} + C_{mr}\left(\frac{1}{\theta^\beta}\left|t^\beta\right|_{W_{j-1}^+}^{W_j}\right)\right)}{nT_p} \quad (2)$$

Where: C_T is as defined in the previous section
C_c is the unit cost of the component (i.e. purchase cost)
C_{ipm} is the cost of performing maintenance
C_{mr} is the cost of performing minimal repair on the component
n is the total number of maintenance that can be performed on the component
β is Weibull shape parameter
θ is Weibull scale parameter
W_{j-1}^+ is the component age at the previous maintenance time
W_j is the component age at the current maintenance time

$$f_j = \frac{T_0}{t_j} f_0 \quad (3)$$

Where: T_0 is the time beyond which the improvement factor begins to drop
f_o is the initial improvement factor of the component.
t_j is any given time in the life of the component

A virtual component with similar data to that used in [2] is also here used for the optimisation. It is assumed that the mean time between failures of the component is same as the Weibull scale parameter. The following data is therefore used for the optimisation.

Improvement factor, $f = 0.985$
Weibull shape parameter, $\beta = 2$
Weibull scale parameter; case i: $\theta = 300$, case ii: $\theta = 7500$
Steady surface hill-climbing move parameter, *move = 10*
Component unit cost = 600
Component cost of maintenance = 160.50
Component cost of minimal repair = 106.0
$T_0 = 120$

The results are presented in two forms; firstly the optimisation was performed on a small population size of 300 time scale (Weibull scale parameter) and secondly on a large population size of 7500 time scale. These two population sizes were considered in order to investigate the characteristic of the steady surface hill-climbing algorithm in optimising maintenance intervals with respect to the component in question. The results obtained are presented next.

5.1. Small Population Size

Using a move parameter of value 10 implies that the number of potential maintenance intervals in the population is 30 (i.e. 300/10). This population is handful and the evaluation of each solution is presented in Table 1 which consists of non optimized solutions. In maintenance terms the move parameter defines the maintenance interval T_p. The results show that the extended useful life of the component is not improved when the maintenance interval is beyond 160. This is because the improvement factor is not constant and therefore a late maintenance is not recommended.

Table 1. Full Solutions for a Small Size Population (time scale of 300)

S/N	T_p	Cost	Extended Life		S/N	T_p	Cost	Extended Life
1	10	18.6082	523.608		16	160	6.05602	418.2
2	20	10.6068	520.825		17	170	5.75848	418.2
3	30	7.95601	518.154		18	180	5.49923	418.2
4	40	6.81896	507.976		19	190	5.27223	418.2
5	50	5.86584	510.79		20	200	5.07264	418.2
6	60	5.35649	510.79		21	210	4.89655	418.2
7	70	5.173	497		22	220	4.74075	418.2

8	80	5.34249	477.3	23	230	4.60259	418.2
9	90	4.81037	487.15	24	240	4.47988	418.2
10	100	4.39409	497	25	250	4.37074	418.2
11	110	5.14219	467.45	26	260	4.27363	418.2
12	120	4.77497	477.3	27	270	4.1872	418.2
13	130	4.47149	477.3	28	280	4.11031	418.2
14	140	4.2181	477.3	29	290	4.04197	418.2
15	150	4.00477	477.3	30	300	3.98133	418.2

Table 2 contains the optimized solutions contained in Table 1. These are solutions that are non-dominated in the population. The steady surface hill-climbing algorithm found 7 solutions that are non-dominated and hence are the optimal solutions or optimal maintenance intervals for the component under a time scale of 300 units.

Table 2. Optimal Solutions for a Time Scale of 300 Units

S/N	T_p	Cost	Extended Life
1	10	18.6082	523.608
2	20	10.6068	520.825
3	30	7.95601	518.154
4	60	5.35649	510.79
5	100	4.39409	497
6	150	4.00477	477.3
7	300	3.98133	418.2

To view the non-dominated solutions in Table 2 in the form of a Pareto frontier, Figure 5 is presented. It can be observed from the figure that the solutions form a curve pattern known as the Pareto front. This means that the rest of the 23 solutions in Table 1 appear in the inside of the Pareto front and are therefore dominated.

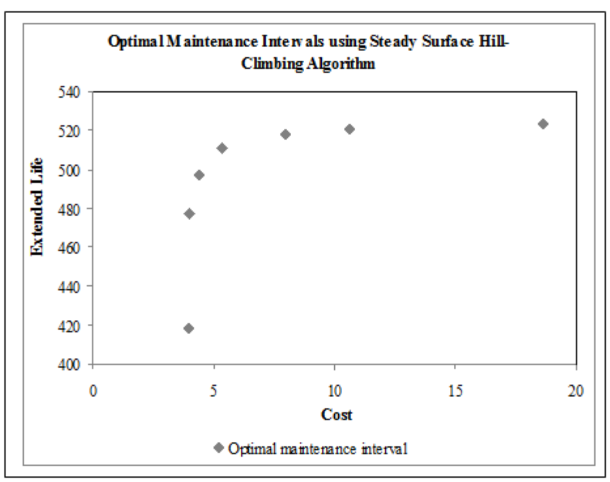


Figure 5. Optimal maintenance intervals from a small population size using SSHC algorithm

5.2. Large Population Size

For a time scale of 7500 units and move parameter of 10, the population size is 750. Due to space limitation, it is not possible to present all the solutions of this population size. The optimisation was performed and the steady surface hill-climbing algorithm found 52 optimal maintenance intervals as seen in Table 3. With a large population size, more optimal solutions are likely to be found. A graphical representation of Table 3 is as shown in Figure 6 which gives the Pareto frontier of the maintenance intervals.

Table 3. Optimal Solutions for a Time Scale of 7500 Units

S/N	T_p	Cost	Extended Life		S/N	T_p	Cost	Extended Life
1	10	16.1515	8102.2		27	270	0.700752	7959.97
2	20	8.12655	8097.56		28	280	0.679739	7955.59
3	30	5.45162	8093.05		29	300	0.638609	7951.05
4	40	4.11447	8088.36		30	310	0.622242	7946.32
5	50	3.31177	8082.08		31	320	0.607246	7941.39
6	60	2.77684	8080.31		32	340	0.576233	7936.25
7	70	2.39491	8071.86		33	350	0.564617	7930.88
8	80	2.10906	8065.21		34	370	0.539278	7925.25
9	90	1.8858	8061.69		35	390	0.517041	7919.34
10	100	1.70714	8059.64		36	410	0.49753	7913.12
11	110	1.56157	8057.99		37	440	0.469684	7906.56
12	120	1.4406	8057		38	460	0.455628	7899.6
13	130	1.3382	8047.15		39	500	0.426082	7892.21
14	140	1.24995	8038.63		40	530	0.409226	7884.33
15	150	1.1725	8031.81		41	570	0.388407	7875.89
16	160	1.10763	8022.05		42	620	0.365654	7866.8
17	170	1.04704	8016.86		43	680	0.342709	7856.95
18	180	0.99604	8008.61		44	750	0.320925	7846.2
19	190	0.948767	8002.77		45	830	0.301295	7834.38
20	200	0.906742	7996.63		46	930	0.281649	7821.25
21	210	0.8693	7990.15		47	1070	0.25942	7806.48
22	220	0.832836	7986.77		48	1250	0.239013	7789.59
23	230	0.802847	7979.71		49	1500	0.219458	7769.89
24	240	0.772734	7976.02		50	1870	0.201291	7746.25
25	250	0.745241	7972.21		51	2500	0.18405	7716.7
26	260	0.723685	7964.19		52	3750	0.171912	7677.3

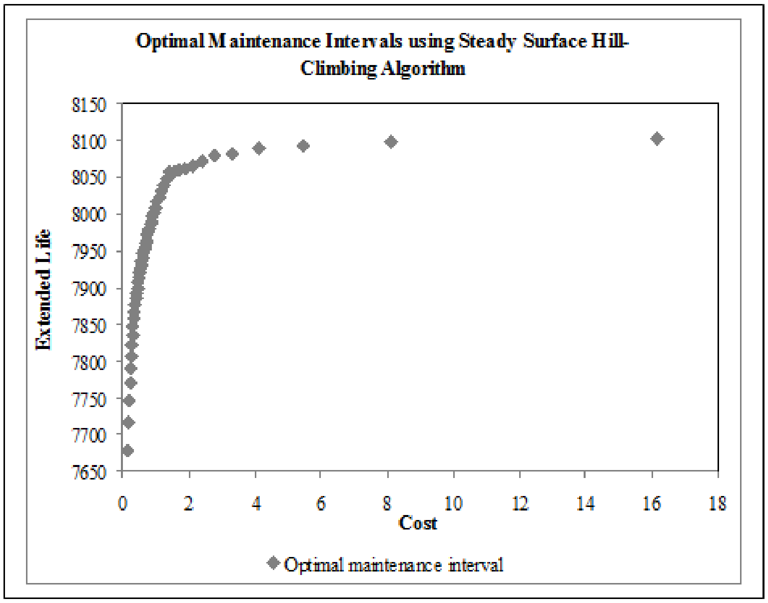


Figure 6. Optimal maintenance intervals from a large population size using SSHC algorithm

As is the case in Figure 5 the rest of the 698 solutions appear in the inside of the curve formed by the optimal solutions shown in Figure 6.

6. Conclusion

The traditional hill-climbing algorithm is known to be suitable for a single objective optimisation problem which produces a single optimal solution. This paper has established a new hill-climbing algorithm called steady surface hill-climbing which is shown to perform multi-objective optimisation and as such produces set of optimal solutions. Also, the steady surface hill-climbing algorithm was demonstrated to extend an earlier work on the longevity of component (IT product) [2]. Hence, optimisation of maintenance intervals for a component was performed with respect to its extended useful life and cost taking into account the effects of maintenance. Optimal set of maintenance intervals for a small and large size population was obtained. The results obtained reveal that it is possible to optimise the maintenance intervals of a component and that the steady surface hill-climbing algorithm is sufficient to achieving this.

References

[1] M.I. Mazhar, S. Kara, and H. Kaebernick, “Remaining Life Estimation of Used Components in Consumer Products: Life Cycle Data Analysis by Weibull and Artificial Neural Networks”, Journal of Operations Management, 25(6), 2007, pp. 1184-1193.

[2] S.H. Nggada, “Reducing Component Time to Dispose through Gained Life”, International Journal of Advanced Science and Technology, Vol. 35, 2011, pp. 103-118

[3] R. Roy, S. Hinduja, and R. Teti, “Recent Advances in Engineering Design Optimisation: Challenges and Future Trends”, CIRP Annals - Manufacturing Technology, 57(2), 2008, pp. 697-715

[4] S. Goyal, and R. Gupta, “Optimization of Fidelity with Adaptive Genetic Watermarking Algorithm using Tournament Selection”, International Journal of Advanced Science and Technology, Vol. 30, 2011, pp. 55-65

[5] F. Musharavati, and A.S.M. Hamouda, "Modified Genetic Algorithms for Manufacturing Process Planning in Multiple Parts Manufacturing Lines", Expert Systems with Applications, 38(9), 2011, pp. 10770-10779

[6] S.H. Nggada, D.J. Parker, and Y.I. Papadopoulos, "Dynamic Effect of Perfect Preventive Maintenance on System Reliability and Cost Using HiP-HOPS", IFAC-MCPL 2010 5th Conference on Management and Control of Production and Logistics, , Coimbra, Portugal, 8 - 10 September 2010.

[7] A. MacFarlane, and A. Tuson, "Local Search: A Guide for the Information Retrieval Practitioner", Information Processing and Management, 2009, (45)1, pp. 159-174

[8] B. Suman, N. Hoda, and S. Jha, "Orthogonal Simulated Annealing for Multiobjective Optimization, Computers and Chemical Engineering", 2009, doi:10.1016/j.compchemeng.2009.11.015

[9] E. Hart, "Non-Deterministic Search Techniques" [online], 2006, Available: http://www.dcs.napier.ac.uk/~benp/ecresources/Search2006.pdf [Accessed 23 September 2011]

Author

Shawulu Hunira Nggada obtained a B. Tech(Hons) in Computer Science from Abubakar Tafawa Balewa University, Nigeria and MSc in Software Engineering from the University of Bradford, UK. He is currently a PhD student at the University of Hull, UK and working on multi-objective optimisation of system maintenance. He is also a Charted IT Professional with the British Computer Society. Some of his research interests are system dependability, software engineering, green computing and programming languages.