A GENERALIZED EXTREMAL OPTIMIZATION-INSPIRED ALGORITHM FOR PREDICTIVE MAINTENANCE SCHEDULING PROBLEMS

Pasquale Arpaia
Dipartimento di Ingegneria, Università del Sannio, CERN European Laboratory for Nuclear Research
Department of Technologies, Group of Magnets, Superconductors and Cryostats
M26220, (30-03-030) CH-1211 Geneva 23, Switzerland

Domenico Maisto
Institute for High-Performance Computing and Networking (ICAR), Italian National Research Council (CNR)
Via P. Castellino 111, 80131 Naples, Italy

Carlo Manna
Dipartimento di Ingegneria Elettrica, Università degli Studi di Napoli Federico II
Via Claudio 21, 80125 Napoli, Italy

Keywords: Artificial Intelligence, Optimization Methods, Maintenance, Scheduling.

Abstract: A bit-encoded heuristic evolutionary optimization algorithm inspired by the Generalized Extremal Optimization method is presented. The proposed evolutionary approach aims at optimizing a predictive maintenance scheduling problem characterized by an analytically intractable objective function. A preliminary comparison with a standard genetic algorithm on a set of high-dimension cases of the considered maintenance problem shows better performance for the proposed approach.

1 INTRODUCTION

Evolutionary algorithms are excellent heuristic methods, inspired by biological evolution, to solve complex optimization problems with analytically intractable objective functions. Although evolutionary-based methods approximate the optimal solution without guaranteeing its optimality, the underlying principles of natural evolution ensure promising results (De Sousa and Ramos, 2002). This turns out to be useful especially in *real-time* complex optimization.

The most popular and used methods are mainly: Genetic Algorithms (GA) (Goldberg, 1989), Simulated Annealing (SA) (Kirkpatrick *et al,* 1983), and algorithms based on Swarm Intelligence, such as Ant Colony Optimization (ACO) (Dorigo *et al,* 1996), and Particle Swarm Optimization (PSO) (Kennedy and Eberhart, 1995), (although the last two are biological inspired heuristics, not considered tightly *evolutionary* by the survey).

However, the aforementioned algorithms in their practical implementation for optimization problems have a problematic feature: the optimal solution is searched through a stochastic process sensitive to a suitable setting of adjustable parameters. A proper setting affects the performance of the algorithms significantly, and in many practical cases this becomes a costly task in itself. Moreover, most of them are population-based, thus their run is time-consuming compared to other algorithms.

By exploiting the Self-Organized Criticality state theory (SOC) (Bak, Tang and Wiesenfeld, 1987) in ecosystems, Boettcher and Percus proposed a novel evolutionary optimization method called Extremal Optimization (EO) (Boettcher and Percus, 2001), successfully applied to complex combinatorial optimization problems. EO method relies on the

Arpaia P., Maisto D. and Manna C..
A GENERALIZED EXTREMAL OPTIMIZATION-INSPIRED ALGORITHM FOR PREDICTIVE MAINTENANCE SCHEDULING PROBLEMS.
DOI: 10.5220/0003080200700076
In *Proceedings of the International Conference on Evolutionary Computation* (ICEC-2010), pages 70-76
ISBN: 978-989-8425-31-7

Bak-Sneppen model (Bak and Sneppen, 1993), a simplified model of natural co-evolution in ecosystems: a number of species in a system evolves to reach the best adaptation; the worst adapted species are *forced* to evolve more quickly to avoid extinction. This mechanism determines an overall adaptation for the ecosystem as a whole.

Beyond these encouraging results, the evolutionary approach proposed in (Boettcher and Percus, 2001) adds two peculiar features: only *one setting parameter* is needed and a single candidate is processed at each iteration. These two aspects are “a priori” advantages with respect to the traditional evolutionary approach (as GA, SA, PSO and so on).

These noteworthy characteristics have encouraged the employment of EO algorithm to tackle different physics issues or engineering applications, particularly hard to face.

Predictive maintenance scheduling belongs to this class of problems; it could be described in this way: an optimal action sequence for maintaining a system in order to avoid potential breakdowns is to be found. The terms *predictive* indicates that some problem parameters cannot be constant during the process, but are continuously updated in *real time*. Thus, the planned schedule (the optimal solution) must to be re-organized for every modification of the examined system state and the constraints of the task. Moreover, such as many maintenance scheduling problems, the corresponding optimization problem is characterized by an analytically intractable objective function to be minimized. Hence, it needs for a heuristic approach to search the optimal solution.

Among the above variations of EO, the Generalized Extremal Optimization (GEO) algorithm (De Sousa, Ramos, 2002) was built to be applied on a wide class of complex problems. Its particularity lies in working on strings composed by bits with “fitness” proportional to the contribution to the quality of the whole solution generated by their mutation.

Following this simple idea, in this paper a GEO application is proposed for the problem of the predictive maintenance. After an outline of the proposed method, preliminary experimental results on a set of analytically intractable scheduling problems are shown in order to highlight better performance than a standard GA.

2 THE PROPOSED METHOD

In the present section, first, a formulation of *predictive maintenance scheduling problem* is detailed and, then, the proposed heuristic algorithm is presented.

2.1 Statement of the Predictive Maintenance Scheduling Problem

2.1.1 Experimental Motivations

The maintenance scheduling formulation proposed in the following is to be faced under the framework of the industrial research project MONDIEVOB (Buildings Remote Monitoring and Evolutionary Diagnostics), granted by *POR 3.17 ICT Regione Campania* (Italy).

The long-term goal of MONDIEVOB is a predictive maintenance tool for processing experimental information acquired from building to be maintained in order to assess reliability and predict possible future failures (Figure 1), by means of algorithms able to predict future status of a machine or a process (Stapelberg, 2009). This predictive information allows *proactive* responsiveness in maintenance decision-making.

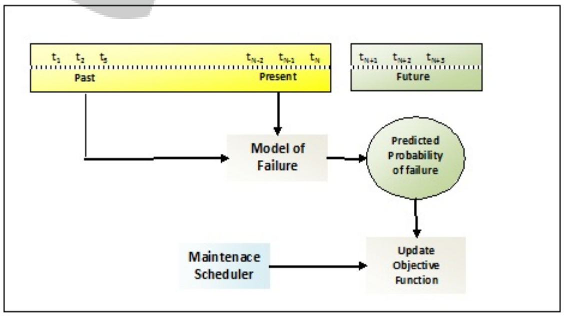

Figure 1: “Model of failure” module predicts probability of failure of the considered system, from past and present data. This predictive information updates the objective function of the maintenance scheduler, in real time.

Essentially, the on-line available information about the status of the monitored systems allows maintenance operations to be anticipated/delayed according to the actual conditions.

In order to accomplish this task, a formulation evaluating different maintenance scenarios by considering the associated cost effects of the resulting maintenance operations and taking into account the current and predicted machine degradation levels has been set up. The cost of maintenance actions, availability and maintenance resource constraints are taken into account.

2.1.2 Evaluation of Maintenance Schedule Effects

The purpose of the method presented in this section is defining a cost function in order to evaluate the effects of any given maintenance operation.

The cost function used here takes into account both: the cost associated to the maintenance action (as, for example, the replacement of a given component), and the cost associated to the system operating in the normal state (as monitoring, inspection and so on).

Let n be the available resources to maintenance operation, and m_i (for $i=1,\ldots, M$) the *i-th* system component that must be maintained (for a total of M components). The function C, representing the total cost of planned maintenance, can be expressed as:

$$C = \sum_{t=1}^{T} \left(\sum_{i \in G_t} \left(a_i + p_i(t) * B_i \right) + \sum_{j \in H_t} \left(k_j + b_j(t) \right) \right) \qquad (1)$$

in which the following notation is used:

- T finite time horizon of planned maintenance
- t *for* $t = 1, \ldots, T$, the *t-th* instant of the time horizon T
- a_i the operating cost of the *i-th* component
- k_j the replacement cost for the *j-th* component
- b_j time dependent maintenance cost of the *j-th* component (Dekker *et al.,* 1997)
- $p_i(t)$ probability of failure of the *i-th* component at the time t
- B_i cost of breakdown of the *i-th* component
- G_t the set of every component not maintained at the time t
- H_t the set of every maintained component at the time t

Moreover, any given planned maintenance evaluated by means of (1) is subject to the following constraints:

(i) Each m_i can be served (maintained) by only one of the n available resources at any time t;

(ii) Each m_i has to be served at least one time instant t during the total time horizon T;

Finally, it should be noted that the probability of failure p_i at the time t could be derived from various deterioration models (Djurdjanovic *et al.*, 2003; Engel *et al.,* 2000; Yu *et al*, 2005), depending on the type of monitored component, and from the nature of information or signals acquired.

2.1.3 Bit encoded Solution

In the present work, each maintenance schedule S (called *sequence*, in the following) evaluated by means of (1) is expressed through a binary string representation as:

$$S = \{ s_{11}, s_{12}, \ldots, s_{1M}; \ldots\ldots; s_{T1}, s_{T2}, \ldots, s_{TM} \} \qquad (2)$$

where s_{ki} is the value of the corresponding bit. For example, $s_{13}=1$, means that the *3-th* component is maintained at the time instant $t=1$.

The sequence representation in (2) is suitable for GEO approach proposed in the present paper and described in the following section.

The maintenance problem is hard to solve even for apparently simple cases (Stapelberg, *2009*), as the time required for computing an optimal solution increases rapidly with the size of the study case.

2.2 Generalized Extremal Optimization for Predictive Maintenance Scheduling

2.2.1 Extremal Optimization

The basic idea of the proposed optimization method is inspired by (Bak and Sneppen, 1993), as a simplified model of natural evolution in ecosystems: a number of species in a system co-evolves to reach the best adaptation; the worst adapted species are *forced* to evolve more quickly to avoid extinction. This mechanism determines an overall adaptation for the entire ecosystem. In fact, according to the Bak-Sneppen model, a macroevolutionary ecosystem pattern is characterized by durable periods of quiescence interrupted by some burst of rapid considerable change. In every part of this pattern, it is possible to observe ecological phenomena of different size, larger ones during the periods of major activity and smaller ones in the more quiet periods. The size and the frequency distributions of these events follow typically a power law that implies 1) dynamics is unique and it underlies both the large and the small events (scale invariance) 2) macroevolutionary behaviour of the global ecosystem emerges spontaneously by local interactions between species.Both features are considered as key issues of a working definition of a particular state, known in statistical physics as Self-Organized Criticality (SOC) (Turcotte, 1999), in which system fluctuates about conditions of marginal stability without intervention of external factors (Bak, Tang and Wiesenfeld, 1987).

Bak-Sneppen model can be simulated through an algorithm in quite few steps. First, for each species, a fitness value in the range [0,1], is drawn from a random uniform distribution. Then, the worst adapted species (the one with least fitness) mutates and a new fitness value is assigned to it. The change of the worst adapted species disfigures the fitness landscape locally, involving also the fitness of its neighbours. For this reason, they are constrained to mutate too, even if they are well adapted. After some iteration, the whole system evolves toward a critical threshold value, bringing all species to a generalized better level of adaptation.

The model described above directly inspires the Extremal Optimization algorithm (Boettcher and Percus, 2001). If we set C as a candidate solution to an examined problem composed of |C| design variables denoted by x_i, its basic heuristic procedure is the following:

1. Initialize the variables x_i of C at will; set C_{best} = C.
2. For the current solution C,
   a) set a fitness F_i to each variable x_i,
   b) find j such that F_j is better than F_i for all i,
   c) choose C′ in a neighborhood N(C) of C so that x_j must change
   d) accept C = C′ unconditionally,

   e) if F(C) - the total fitness of the solution C - is better than F(C_{best}) then set C_{best} = C.
3. Repeat step (2) as long as desired.
4. Return C_{best} and F(C_{best}).

Although the above procedure shows good performance in problems where there exist many neighbourhood configurations of C, otherwise it leads to a deterministic process that damages the search of the optimal solution. To avoid this, the algorithm was modified in some steps by introducing into a single parameter (Boettcher and Percus, 2001).

In particular, the steps 2.b and 2.c, have been modified as follow: in step 2b the |C| variables x_i are ranked so that to the variable with the worst fitness is assigned rank 1, and to the one with the best fitness rank |C|. Each time the algorithm passes through step 2c a variable is chosen to be mutated according to a probability distribution of the *k* ranks, given by:

$$P(k) \approx k^{-\tau} \quad , \quad 1 \le k \le N, \qquad (3)$$

where τ is a positive setting parameter. By means of the parameter τ, the algorithm can choose any variable to evolve, although the most probable variables are those with worst fitness.

2.2.2 Generalized Extremal Optimization and his Application

Unfortunately, EO application to a broad class of problems is limited by some drawbacks. One of these consists in giving a general definition for the fitness of the single species, and this means that different problems have different ways to assign the fitness to the variables (Boettcher and Percus, 2001).

To clear the hurdle, De Sousa and Ramos devised a generalization of the EO called Generalized Extremal Optimization (GEO) (De Sousa and Ramos, 2002) capable to tackle either continuous, discrete or integer variables. In GEO, the variables of the optimization problem are arranged in a string similar to a GA chromosome, as it is shown in Figure 2.

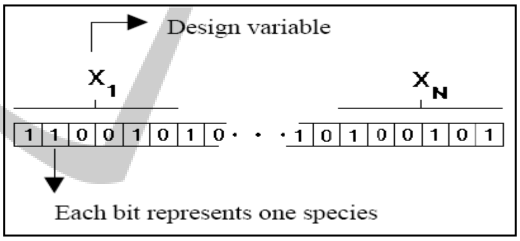

Figure 2: An example of the GEO encoding: *N* design variables of 6 bits. Each bit is considered as a species (De Sousa and Ramos, 2002).

This section deals with the details of a GEO approach to the predictive maintenance scheduling problem solving - that is also a direct way to illustrate the GEO procedure.

The goal of the proposed method is to find the best sequence, expressed as in (2), that minimizes the objective function (1) for the problem described in the section 2.1.

Let us consider a sequence (i.e., a maintenance schedule); as aforementioned, a sequence can be encoded in a binary string, denoted by *S* of length (*M*T*) by means of the representation shown in (2). This manner to express a sequence is particularly suitable to be faced with a GEO. Indeed, in analogy to what EO algorithm does, GEO works on a population (configuration) by muting, generation after generation, a single species (component) and by estimating the obtained candidate solution, with the aim to reach the optimal one.

Thus, if one assumes that every representation bit encodes a single species then an entire population can be expressed by means of a binary string, hence by a sequence in the form (2) too. Figure 3 illustrates the correspondence between bit-species and sequence-population in a visual way.

For the above reasons, a GEO algorithm can straightly work on a sequence *S* by evaluating the candidate solution to the considered maintenance problem through the cost function (1). This means that the lesser is the cost of the sequence the better is the scheduling. Differently than the EO algorithm, at each bit (species) is assigned a fitness value proportional to the decrease of the function (1) computed for the sequence with that bit flipped (i.e., mutated from 1 to 0 or vice versa).

Then, each bit is ranked, such that: to the one with the least fitness is assigned rank 1, while to the one with the best fitness rank N.

Subsequently, a new sequence is generated by flipping a bit chosen according the probability law (3) defined on the rank set.

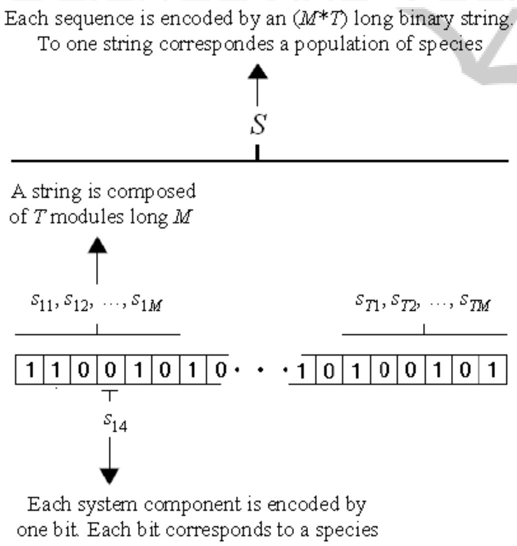

Figure 3: A candidate solution in our GEO approach is a sequence *S* (evaluated by (1)), composed of *(M*T)* bits, as defined in (2). In this example, M=6 components are maintained by *n*=3 resources in the time horizon T.

This iterative process halts after a prefixed number of generation, and it returns the best sequence S_{best} which minimizes the objective function (1).

The proposed procedure is described by the following pseudo-code:

1. Initialize a bit sequence S (with size M*T) randomly and evaluate the objective function C (as in **(1)**);
   Set: S_{best} = S and C_{best} = C(S);
2. For each generation:
   a) For each bit *i* of S:
      - Change the bit *i* (from 1 to 0, or vice versa) and evaluate the cost $C(S_i)$ (as in **(1)**) for S_i;
      - Evaluates the fitness of bit *i* as $\Delta C(S_i) = C(S_i) - C_{best}$
      - Restore the bit i to its previous value.

   b) Sort $\Delta C(S_i)$ in ascending way;
   c) Choose the bit to change with probability **(3)**;
   d) Set S = S_i and C = $C(S_i)$;
   e) If C < C_{best} then set C_{best} = C, and S_{best} = S;
3. Return S_{best} and C_{best}

It is worth underlining that, as regard to the traditional evolutionary algorithms (GA, SA and so on), the present procedure has twofold advantage: (i) there is *only one adjustable parameter* τ, so it simplifies the setting task and (ii) the entire evolution is made on *one configuration solution S* at the time, unlike the traditional evolutionary population based algorithm, and this entails lesser computational costs and a better memory management.

3 EXPERIMENTAL RESULTS

Preliminary experimental tests are carried out in order to validate the effectiveness of the proposed GEO algorithm in comparison with a standard GA (SGA) on four high dimension scheduling cases of the problem described in section 2.1. The problem parameters are reported in Table 1, while the parameter settings for both algorithms are reported in Table 2.

Table 1: Problem parameters: the bit encoded solution length, as defined in (2), is evaluated by means of *(M*T)* bits.

Problem parameters				
#	*M*	*n*	*T*	*Solution length (2)* [bit]
1	*6*	*2*	*6*	*36*
2	*8*	*3*	*8*	*64*
3	*10*	*4*	*10*	*100*
4	*10*	*4*	*15*	*150*

As it should be noted from Table 2, the population size for both algorithms are not reported. This because the proposed GEO works only on 1 individual and performs a number of evaluations as the solution length. Therefore, for example, in the problem #1 the proposed GEO performs 36 evaluations at each iteration, while, for the problem #3 it performs 100 evaluations at each iteration. Therefore, in order to compare the proposed GEO and SGA, the population size of the last algorithm has been set to: 36 for the problem #1, 64 for the problem #2, 100 for the problem #3 and, finally, was 150 for the problem #4.

For each problem, it was performed 50 runs for both algorithms.

In Table 3 and Table 4 are reported the preliminary experimental results.

In particular, Table 3 shows a comparison between the costs of the best solutions (mean value and standard deviation) achieved for GEO and SGA. As one can see, both algorithms have the same performance on the first two cases (#1 and #2), but GEO outperforms SGA better and better while increasing the size of the sequence.

However, in Table 4, the difference between the algorithm presented in this work and SGA is noticeable.

Table 2: Parameter settings for the GEO application and the standard GA.

Proposed GEO	GA	
τ = 0,75	*Mutation mechanism*	*Uniform*
	Crossover mechanism	*Single point*
	Crossover fraction	*0.8*
	Selection mechanism	*Roulette*

In particular, the proposed approach obtains the best solution in lesser number of iteration on the average, highlighting appreciable results.

Table 3: Best solution achieved (mean and standard deviation) by means of the GEO algorithm and the standard GA, for the 4 scheduling cases of Table 1.

Comparison test: best solution cost				
#	*Mean*		*Standard deviation*	
	GEO	*GA*	*GEO*	*GA*
1	70	70	0	0
2	104	104	10^{-3}	10^{-3}
3	164,04	170,1	3,97	4,06
4	219,54	400,01	4,95	4,12

Table 4: Number of iteration on average and standard deviation to achieve the best solution by means of the GEO application and a standard GA, for the 4 scheduling cases of Table 1.

Comparison test: number of iteration				
#	*Mean*		*Standard deviation*	
	GEO	*GA*	*GEO*	*GA*
1	65,12	3876	60,06	2177,16
2	1804,12	3636,6	1589,75	1961,98
3	3081,18	5306,5	1979,98	2062,98
4	8513,54	16667,3	5761,65	9164,24

4 CONCLUSIONS

In the present paper, a Generalized Extremal Optimization (GEO) based algorithm for a predictive maintenance scheduling problem has been proposed.

Preliminary tests on a set of high dimension scheduling problems for the GEO algorithm compared with a standard GA shown encouraging performance of the proposed approach.

In particular, the proposed GEO reaches the best solution in lesser number of iteration on average, compared with the standard GA.

Finally, as previously mentioned, the proposed GEO has a peculiar feature: *a single candidate is processed at each iteration.*

For this reason, a comparison between an evolutionary algorithm having the same feature (as, for example, Simulated Annealing) and the proposed one, should be carried out in the future research.

REFERENCES

Bak, P. and Sneppen, K., 1993, Punctuated Equilibrium and Criticality in a Simple Model of Evolution. *Physical Review Letters, Vol. 71, Number 24, pp. 4083-4086.*

Bak, P., Tang, C. and Wiesenfeld, K.,1987, Self-organized criticality: an explanation of 1 / *f* noise. *Physical Review Letters,* 59: 381–384

Boettcher, S. and Percus, A. G, 2001, Optimization with Extremal Dynamics. *Physical Review Letters, Vol. 86, pp. 5211-5214.*

Dekker R. Wildeman R. E., Van der Duyn Shouten F. A., 1997, A Review on multi-component maintenace models with economic dependence, *Mathematical Methods on Operetion Research. 45: 411-435*

De Sousa, F., Ramos, M. R., 2002. Function Optimization using Extremal Dynamics. In *4th Int. Conf. On Inverse Problem in Engineering, Rio de Janeiro, Brazil*

Djurdjanovic, D., Lee, J., & Ni, J., 2003, Watchdog agent—an infotronics based prognostics approach for product performance assessment and prediction.

International Journal of Advanced Engineering Informatics, Special Issue on Intelligent Maintenance Systems, 17(3–4), 109–125.

Dorigo, M., Maniezzo, V., Colorni A., 1996,, The Ant System: Optimizing by a colony of cooperating agents, *IEEE Trans. Syst. Man Cybern-Part B, vol. 26, n°1, pp 1-13*

Engel, S. J., Gilmartin, B. J., Bongort, K., Hess, A., 2000, Prognostics, the real issues involved with predicting life remaining. *Proc. Of the IEEE Aerospace Conference Proceedings, 6, 457–469.*

Goldberg, D. E., 1989, Genetic Algorithms in Search, Optimization, and Machine Learning, *Addison-Wesley Publishing Company.*

Kennedy, J., Eberhart, R. C. 1995 Particle Swarm Optimization, *In Proc. of the 1995 IEEE International Conf. on Neur. Networks, vol. 4, pp. 1942-1948, IEEE Press, Piscataway, NJ*

Kirkpatrick, S., Gellat, C. D. and Vecchi, M. P., 1983, Optimization by Simulated Annealing, *Science, Vol. 220, Number 4598, 1983, pp. 671-680.*

Stapelberg, R. F., 2009, Handbook Reliability, Availability, Maintainability, and Safety in Engineering Design, *Springer – Verlag 2009*

Yu, G., Qiu, H., Djurdjanovic, D., Lee, J., 2005, Feature signature prediction of a boring process using neural network modeling with confidence bounds, *International Journal of Advanced Manufacturing Technology.* Turcotte, D.L., 1999, Self-organized criticality. Reports on progress in physics, vol 62, n° 10